

Coastal Community Climate Change Adaptation Framework Development and Implementation

**Master's Program in System Science
Faculty of Graduate and Postdoctoral Studies
University of Ottawa**

**Mingliang Lu
Supervisor: Dan Lane, Telfer School of Management**

Thesis

June 2013
© Mingliang Lu, Ottawa, Canada, 2014

Table of Contents

Abstract.....	7
1 Introduction	8
1.1 Background and Motivation	8
1.2 Research Questions and Objectives	10
1.3 Plan for the Thesis	11
2 Literature Review.....	13
2.1 Community Profile.....	13
2.2 Threats and Vulnerabilities	15
2.3 Adaptation Frameworks	18
2.3.1 ICLEI.....	18
2.3.2 FCM.....	19
2.3.3 ICSP	20
2.3.4 MCCAP (Municipal Climate Change Action Plan)	20
2.4 Action Planning.....	21
2.4.1 FEMA	21
2.4.2 EMO.....	22
2.4.3 Business Planning.....	23
2.5 Information Technology and System.....	24
2.5.1 Google Crisis Response.....	24
2.5.2 Ushahidi	25
2.6 Community Applications.....	27
2.6.1 Charlottetown's ICSP	27
2.6.2 Isle Madame MCCAP	27
2.7 Summary	28
3 Methodology.....	29
3.1 Process Modeling.....	29
3.1.1 Community Profile.....	30
3.1.2 Threats and Vulnerabilities	33
3.1.3 Adaptation Team	37
3.1.4 Action Planning.....	40
3.2 System Analysis.....	49
3.3 Information Process	52
4 Implementation	56
4.1 Community Adaptation Website Development.....	56
4.2 Community Adaptation Mobile Application Design	59
4.2.1 Functions.....	62
4.2.2 Data Management	73
4.2.3 User Interface Design	76
4.3 Validity and Verification	89
4.3.1 Programming.....	90
4.3.2 iOS Device Testing	94
5 Simulation	96
5.1 Community-Wide Simulation	96
5.2 Community Area Simulation.....	106

6	Conclusions.....	112
6.1	Summary	112
6.2	Future Study.....	115
6.2.1	Questions and Concerns.....	115
6.2.2	Recommendations.....	117
7	Bibliography.....	119
Appendices		122
Appendix A: Charlottetown Green House Gas Emission Inventory Summary.....		122
Appendix B: Comparison of ICLEI and MCCAP Adaptation Framework		124
Appendix C: Charlottetown ICSP Template		128
Appendix D: Richmond County MCCAP Template.....		132
Appendix E: Community Adaptation Mobile Application Code		145

List of Figures

Figure 2.1: Vulnerability (Australian, 2005)	17
Figure 2.2: Five-Milestone Framework (Source: ICLEI, 2012)	18
Figure 2.3 (Source: Nova Scotia, 2011).....	23
Figure 2.4: Google Crisis Response (Source: http://google.org/publicalerts)	25
Figure 2.5: Ushahidi Mobile Application (Source: http://download.ushahidi.com)	26
Figure 2.6: Ushahidi Mobile Application (Source: http://download.ushahidi.com)	26
Figure 3.1: Climate Change Adaptation Process	30
Figure 3.2: Community Profile-Elements of the Environmental Pillar	31
Figure 3.3: Community Profile-Elements of the Economic Pillar.....	32
Figure 3.4: Community Profile-Elements of the Social Pillar	32
Figure 3.5: Community Profile-Elements of the Cultural Pillar	33
Figure 3.6: Community Threats Framework	36
Figure 3.7: Community Adaptation Team Hierarchy	37
Figure 3.8: Reoriented Adaptation Team Significance.....	40
Figure 3.9: Action Planning Aspects: Before, During, and After.....	41
Figure 3.10: Before Events: Training Aspect	42
Figure 3.11: Before Events: Notification Aspect.....	43
Figure 3.12: Before Events: Preparedness Aspect	44
Figure 3.13: During Events: Emergency Aspect	45
Figure 3.14: During Events: Communication Aspect.....	46
Figure 3.15: During Events: Transportation Aspect	47
Figure 3.16: After Events: Supply Aspect	48
Figure 3.17: After Events: Rebuilt Aspect.....	48
Figure 3.18: After Events: Recover Aspect	49
Figure 3.19: Current Centralized Local Adaptation Situation	50
Figure 3.20: Improved Local Community Adaptation and Response Situation	50
Figure 3.21: Community Member Profile Information	53
Figure 3.22: Threats Information	53
Figure 3.23: Team Information.....	54
Figure 3.24: Action Planning Information.....	54
Figure 4.1: Joomla Control Panel	57
Figure 4.2: Community Adaptation Website Structure	58
Figure 4.3: Community Adaptation Website	58
Figure 4.4: Features of Apple's iPad Mobile Application System.....	60
Figure 4.5: Mobile Application System Components for Development and Implementation	61
Figure 4.6: Customizable Community Mapping System.....	64
Figure 4.7: Community Location Layers.....	65
Figure 4.8: Community Vulnerable Areas Indication.....	66
Figure 4.9: Location and Status Sharing Window	70
Figure 4.10: Advisory Places Controller Window.....	71
Figure 4.11: Advisory Areas Controller Window.....	72
Figure 4.12: Mobile Application Functions.....	73
Figure 4.13: Annotation Button Action	78

Figure 4.14: Vulnerable Areas Button Action	79
Figure 4.15: Advisory Information Button Action	80
Figure 4.16: User Locations Button Action	81
Figure 4.17: Location Sharing Button Action.....	82
Figure 4.18: Editing Map Button Action	83
Figure 4.19: Editing Advisory Places Button Action	84
Figure 4.20: Editing Advisory Areas Button Action	85
Figure 4.21: Home View.....	86
Figure 4.22: Map View	87
Figure 4.23: Login View	88
Figure 4.24: Administrator View	89
Figure 5.1: Simulation Initial Location and Status Indicator for 29 Richmond County Users	98
Figure 5.2: Receiving More Information From Community Users	99
Figure 5.3: Status Changing in Community Users	100
Figure 5.4: Changing Status into Under Help.....	101
Figure 5.5: Posting Advisory Area Information	102
Figure 5.6: Changing Status Back to Normal	103
Figure 5.7: Community Users Status Update	104
Figure 5.8: Event Finished	105
Figure 5.9: Little Anse Road Closure event (top left in photo). (Source: Apple Map) ..	107
Figure 5.10: Community Users Help Status at Step 2. (Source: Apple Maps).....	108
Figure 5.11: More Community Users Receiving Help from “Helpers” at Step 3. (Source: Apple Map)	109
Figure 5.12: Most Community Users Are Out of Trouble (Source: Apple Map).....	110
Figure 5.13: Event Completed (Source: Apple Maps).....	111

List of Tables

Table 1: Green House Gas Emission By Sector	122
Table 2: Green House Gas Emission By Source	123
Table 3: Resident Green House Gas Emission	123
Table 4: Commerce Green House Gas Emission.....	123
Table 5: Industry Green House Gas Emission	123
Table 6: Mitigation Comparison	125
Table 7: Adaptation Comparison	127
Table 8: Environmental Pillar. (Source: ICSP, City of Charlottetown, 2010)	129
Table 9: Economic Pillar. (Source: ICSP, City of Charlottetown, 2010)	130
Table 10: Social Pillar. (Source: ICSP, City of Charlottetown, 2010)	131
Table 11: Cultural Pillar. (Source: ICSP, City of Charlottetown, 2010)	131
Table 12: Suggested Committee Structure. (Source: Staff Directory. Richmond County's Website)	132
Table 13: Hazards & Issues. (Source: December 2011, C-Change Presentation)	133
Table 14: Affected Locations in Isle Madame. (Source: 2011-2012, Isle Madame Research Project)	133
Table 15: Facilities and Infrastructure. (Source: Public Works, Richmond County's Website)	133
Table 16: Environmental Pillar. (Source: Richmond, 2010)	134
Table 17: Economic Pillar. (Source: Richmond, 2010)	139
Table 18: Social Pillar. (Source: Richmond, 2010)	142
Table 19: Cultural Pillar. (Source: Richmond, 2010)	143
Table 20: Project Priorities. (Source: Richmond, 2010)	144

Abstract

As the impacts of climate change become more severe, coastal communities are required to prepare plans for adaptation to the invasive environmental changes. A well-prepared adaptation plan can effectively reduce the overall risks of coastal communities. However, a plan is not the final solution for the climate change on coastal communities. How to take the plan into action and implement it in the local communities and find the opportunities for the enhanced preparedness and development of coastal communities is the primary consideration of this thesis research. Many organizations are engaged in developing adaptation tools and guidebooks. For completing their adaptation plans, communities need to develop clear, operational, action plans, and discover the opportunities to enhance the sustainability of coastal communities. To make coastal communities more sustainable in the face of the changing climate, the public's attention and community participation is critical. The purpose of this study is to develop an adaptation framework and action plan process system for coastal communities and at the same time, provide the general public with an enhanced opportunity to contribute their understanding about what is being done for their coastal community around them and how to react when an event happens. The research is applied to the coastal communities of Richmond County, Cape Breton, Canada as a case study. The result of the work develops an adaptation "Action Plan" website for Richmond County. The website features the development, application, and simulation of a mobile communication "Action plan" application designed and implemented with the action website along to provide coastal community with communication options that exploit the local community network and enhance the community's capacity for climate change adaptation. The emergency response community mobile app and the accompanying website are models for other communities especially those that from the coastal communities in Canada and the Caribbean as part of the C-Change ICURA project to which this research is affiliated.

1 Introduction

This document presents thesis research carried out in partial fulfillment of the M.Sc. requirements of the Systems Science program. This thesis is duly submitted to the University of Ottawa's Faculty of Graduate and Postdoctoral Studies.

1.1 Background and Motivation

The increasing impacts of climate change challenge the vulnerability of Canadian coastal communities. Severe storms, flooding from storm surge, and coastal erosions happen more frequently than in the past and represent a real threat to coastal community and additional costs to provide emergency preparedness and local government services. Since Canada has the longest coastline in the world with abundant coastal resources on three oceans and the Great Lakes, the impact of risk-laden climate changes puts these areas into potential danger and directly affects the local coastal economies with respect to property values, opportunities for commerce and investment.

The IPCC's (Intergovernmental Panel on Climate Change) "Climate change 2007: Synthesis Report" presents strong evidence from scientific research to show that the pattern of global climate is changing and impacts are becoming more and more obvious. The emission of greenhouse gas produced by burning carbon-based fuel resources, contributes to global warming and climate change. The rising temperatures lead to melting of land-based glaciers and thermal expansion lead to ice melting, which in turn leads to global sea level rise. Abnormal precipitation in some areas and drought in other areas coupled with severe weather events everywhere are more immediate evidence of the changes in the climate around the world.

In response, effective adaptation plans for communities are expected to contain two parts: (1) adaptation to the pending environmental changes, and (2) mitigation of greenhouse gas emission and management of carbon in the atmosphere. In 1997, the development of the Kyoto protocol aimed to stabilize the emission of greenhouse gas at a certain level in an effort to reduce climate change attributed to human activities (United Nations

Framework Convention on Climate change, 1998). In order to achieve the Kyoto objectives, some Canadian communities prepared greenhouse gas (GHG) emission inventories and set up reduction goals in accordance with developing strategies to combat the changing environment (e.g., the Federation of Canadian Municipalities' Partners for Climate Protection program, FCM 2009)

In 2007, at the UNFCCC (United Nations Framework Convention on Climate Change), Communities of Practice meeting, COP13, the Bali Road Map focused for the first time on building an adaptation action plan as a means to reach the Kyoto objectives that were, by this time, becoming more and more elusive. Following the "adaptation" lead of the (Carpenter, 2007), ICLEI (Local Governments for Sustainability) and FCM (Federation of Canadian Municipalities) developed their five-milestone framework that combined both mitigation and adaptation.

Recently, many Canadian communities, spearheaded by the Province of Nova Scotia, are now engaged in applying the MCCAP (Municipal Climate Change Action Plan) program, which was adapted from the ICSP (Integrated Community Sustainability Plan) of 2009 as a means of municipalities receiving funds as part of the Agreement on The Transfer of Federal Gas Tax Revenues (Canada, 2006) with the provinces. In 2009, the C-Change ICURA (International Community University Research Alliance) research group was established as a coastal climate change adaptation project to assist Canadian and Caribbean coastal communities towards developing environmental awareness, enhancing adaptive capacity, and becoming more resilient in the development of operational climate change action plans.

To date, action planning for climate change exists mainly as an uncoordinated, under-resourced, and non-prioritized activity that must be included in the workloads of coastal communities small and large. Following the events of natural disasters such as Hurricane Sandy in October 2012, this thesis expresses urgency in the need for coastal communities to prioritize their local climate action plans. The following section defines the objectives of this research to redress the issue and provides a research program for operational

aspects of action planning for coastal communities. The work is applied to specific communities as part of the C-Change project toward enhancing community resilience in the face of changing coastal climates.

1.2 Research Questions and Objectives

As noted above, some Canadian coastal communities have participated in programs to prepare a climate change adaptation action plan. These plans are typically divided into two parts: (1) mitigate GHGs, and (2) adaptation. To this end, communities have prepared GHG emission inventories and set up reduction targets as “first steps” in the developing their climate change action plans.

Communities also need to perform adaptation planning. Adaptation is typically seen as a long-term and strategic effort, but one that is very necessary. The questions asked in this thesis are about what are the contents of an adaptation plan, and how can an adaptation framework be built to describe the problems communities are currently facing for effective action planning. This research considers that community action plans need to be embraced by engaging all community members. As well, this requires an information system designed to communicate and integrate the adaptation framework elements with and operational action planning process for local governments as well as individual community members’ use.

The specific questions of this thesis research are as follows:

- (1) How can the current status and the vulnerability of a coastal community described?
- (2) What are the elements of an effective community adaptation plan?
- (3) What are effective tools and processes required for operational action plan implementation?

To examine these research questions, the proposed objectives of this research are:

- (1) To develop a framework for representing the profile and vulnerability of the coastal community profile.

- (2) To construct a framework for defining the elements and actionable items for an effective community climate change adaptation and action plan.
- (3) To implement specific elements including a community website and a local community mobile application as part of the community's strategic plan and its operational action plan in the event of emergencies to engage all community members.

1.3 Plan for the Thesis

The outline of the thesis document to follow is described below:

In the current section, Chapter 1 introduced the background and motivation of the problem and notes the need for the development of coastal community adaptation plans from the lessons learned in the work of international organizations. This chapter also presented the questions and objectives for this thesis research.

Chapter 2, the literature review, reviews the components of building a community profile and the threats and vulnerable places related to climate change. This chapter also introduces recent adaptation elements, which are developed by international and domestic (national) organizations for local communities and municipalities.

Chapter 3 presents the research methods including the proposed process to develop an adaptation framework and action plan community website and mobile application and outlines specific strategic and operational items.

Chapter 4 provides the results of this study including application, and testing of the information system-based action for a coastal community, including the development and implementation of a municipal website and a mobile device application for communicating advisories and community members status for action in emergency situations.

Chapter 5 prepares two simulation plans for both (1) community wide and (2) specific area emergency event response using the mobile application.

Chapter 6 summarizes the overall thesis research work and provides summary, conclusions and recommendations for future research.

The research bibliography and the set of appendices complete the contents of the thesis.

2 Literature Review

The review of the relevant literature is provided in sections in this chapter. The following section, (2.1) presents the community profiling information according to the four community pillars: environmental, economic, social and cultural. In section (2.2), the community climate change related threats and vulnerability research are presented. Section (2.3) discusses adaptation frameworks from international examples and from Canada. After that, in section (2.4), action planning structures and organizations are introduced. Then in section (2.5), in the final section of this chapter (2.6) provides climate change adaptation applications from the C-Change project, namely, the C-Change communities of Charlottetown and Isle Madame, Richmond County.

2.1 Community Profile

The purpose of defining the community profile is to develop a clear understanding of the situations and status of coastal communities. Many municipalities have already collected useful information and have had experience in monitoring their changing environmental conditions, however, how to organize and use this information to build a community adaptation profile is important. The community profile includes the background information, threats, and vulnerable places with respect to pending climate change.

The ICSP (Integrated Community Sustainability Plan) prepared by Canadian communities in 2009-2010 developed long-term strategies to achieve adaptation goals by using a four-dimensional set of community profile pillars, which are the environmental, economic, social and cultural background information.

Environment is a major aspect of the community profile, because some elements in the environment are directly under threats, such as ecosystem, weather, and resources, the impacts are huge and hard to prevent, but communities can reduce the impacts to environment caused by human activities by “reducing the reliance on fossil fuels and the production of greenhouse gas emissions, reducing use of chemicals and other manufactured substances that can accumulate in the environment, and the protection of

ecosystem integrity and diversity” (Newfoundland and Labrador, 2009, p.6). Historical climate data such as precipitation, temperature and wind speed are required when municipalities want to make an adaptation plan depending on the seasonal trends of the local climate. Information about natural resources and the ecosystem are also important components of the community environmental pillar.

The economic pillar presents municipalities’ economic activities and lists the assets, which belong to local governments. For coastal communities, major economic activities are related to the areas of fisheries, tourism or local business. In this pillar, more information should be collected based on the areas, which contribute the most part of income to local communities. Community assets normally include the properties and infrastructures of local government. Every year, it costs millions of dollars for coastal communities to maintain and repair infrastructures damaged by severe weather events. The purpose of making a community economic profile is to “promote economic development that provides opportunities for people of different incomes and skills, promotes a better quality of life and protects the environment while maintaining sufficient revenues for the municipality” (Newfoundland and Labrador, 2009, p.7).

The social pillar focuses on the community’s services and residents’ needs. To make a sustainable society, some conditions need to be met :

- “ 1. The basic needs of citizens with respect to food, shelter, education, work, income and safety are met.
2. Family friendly environment are developed through access to childcare, youth programs and other family services.
3. Access to opportunities, which are distributed equitably and fairly across society.
4. There is an environment of hospitality, wherein citizens live together harmoniously and in mutual support of each other.
5. Individuals have the opportunity and are encouraged to participate in civic processes.” (Newfoundland and Labrador, 2009, p.7)

A community social pillar based on these conditions assists municipalities easily to

determine what needs to be improved to provide better services to citizens.

The last aspect of community profile is culture. It is unique for each local community and hard to define. Basically, historical sites and heritage are the visible components of community culture and easy to manage. However, other components such as art, history and aboriginal are different among communities.

“Many municipalities have invested in arts and heritage as important element of their revitalization and renewal strategies. Cultural tourism is also on the rise and is increasingly linked with community branding and economic development initiatives” (Newfoundland and Labrador, 2009, p.6)

2.2 Threats and Vulnerabilities

In 2012, ICLEI reported that:

“Many of the most severe and costly climate change impacts will be associated with projected increases in the frequency and magnitude of extreme climate events and associated natural disasters. These events include flooding due to high-intensity rainfall and storm surges, ice and windstorms, hail, heat waves and drought.” (ICLEI, 2012, p.12)

In order to reduce the impacts caused by extreme weather events, coastal communities need to identify clearly how they affect the functioning of the community. For example, the threats to communities' environment could be “impacts on hydrology, changes in biodiversity and impacts on agriculture” (Bizikova, 2008). As Gupta also noted: “The threats, however, are not just environmental; climate change effects have been seen in social, political, and even economic contexts” (Gupta, 2007).

Since infrastructure is one of the most important assets for coastal communities, how to prevent and reduce the threats caused by climate change to these infrastructures become local governments' important tasks. The types of infrastructures can be considered as:

“transportation; energy generation and transmission; water, sewer, and telecommunications; coastal defense”, and the threats associated with climate change to these infrastructures are:

“coastal flooding and inundation from sea level rise; changes in patterns of water availability; effects of higher temperature on operating costs, including effects in temperate areas and areas currently characterized by permafrost conditions; and demand-included effects.” (Neumann and Price, 2009, p.1)

The complexity and difficulty of climate change issues makes it hard to identify how and where the threats would affect the community. Geographic considerations are essential factors, and can help communities have a better understanding of how to analyze and manage the complex issues of the changing coastal climate. GIS (Geographic Information Systems) is a powerful tool to explore these critical problems (Dangermond and Artz, 2010, p.1).

Adger states “Vulnerability is the state of susceptibility to harm from exposure to stresses associated with environmental and social change and from the absence of capacity to adapt” (Adger, 2006, p.1).

Similarly, others state that,

“Key vulnerabilities are associated with many climate-sensitive systems, including food supply, infrastructure, health, water resources, coastal system, ecosystems, global biogeochemical cycles, ice sheets and modes of oceanic and atmospheric circulation.”

(Patwardhan, Semenov and Schneider, 2007, p.13)

“Analysis starts with an assessment of a municipality’s vulnerability to the current climate, informed by knowledge of how climate events have impacted the community in the past, the actions that were taken in response to those impacts and how successful those actions were.”

(Lemmen and Warren, 2007, p.5)

Understanding climate vulnerability can also provide communities with a basic idea about adaptation possibilities (Richardson, 2010). The approaches to analyze community vulnerability normally focus on the physical indicators such as how often an extreme weather event happens and what impacts it may have on the community. However, the development of a social indicator considers more about reducing the severe event impacts on people and the community economy (Adger, 2005, p.2). Based on this idea, a new vulnerability socio-economic indicator (Adger and Brooks, 2004, p.6) is developed.

Socio-economic impacts arise from social vulnerability in the community. Further, “Social vulnerability to climate change is a function of exposure, sensitivity, and adaptive capacity.” (Wongbusarakum and Loper, 2011, p.8)

The relations among these components are,

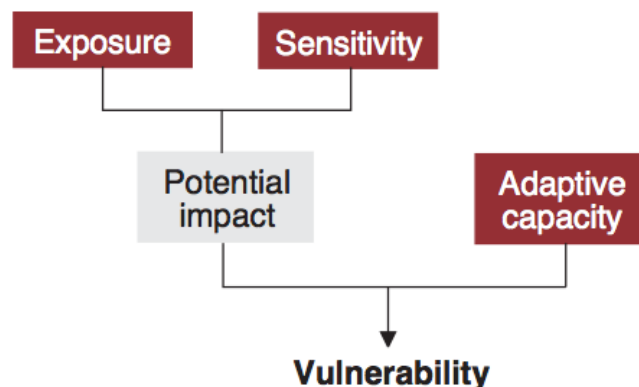


Figure 2.1: Vulnerability (Australian, 2005)

2.3 Adaptation Frameworks

As discussed above, evidence shows that the climate is changing and impacts are obvious. In order to reduce these impacts, communities around the world need to learn how to adapt to the pending change. Many organizations and research groups have done work to help coastal communities toward developing an adaptation plan. The following sections introduce four organizations' adaptation planning frameworks, in which global and Canadian communities have taken part. These include (1) ICLEI, (2) FCM, (3) ICSPs, and (4) MCCAP, as described in further detail below.

2.3.1 ICLEI

ICLEI (Local Governments for Sustainability) is an international association of local governments as well as national and regional local government organizations who have made a commitment to sustainable development. "ICLEI's five milestones for Climate adaptation methodology provide a structured approach to adaptation planning which moves participating local governments through a series of progressive steps." (ICLEI, 2012, p.6)

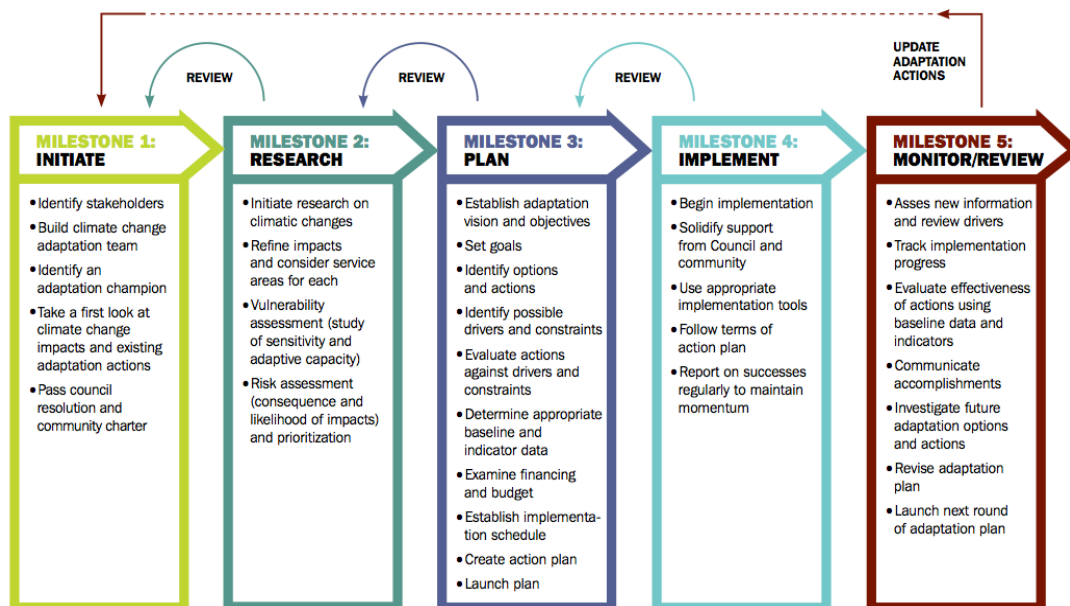


Figure 2.2: Five-Milestone Framework (Source: ICLEI, 2012)

The five-milestone adaptation framework developed by ICLEI is designed to provide local government tools and resources to fulfill implementation. It is also pointed out that

“The multifunctional nature of these resources is in response to the fact that while adaptation planning can be a long-term and complex process, there are exercises that can be done in the short term to support a longer-term process.” (ICLEI, 2012, p.2)

2.3.2 FCM

The Federation of Canadian Municipalities (FCM) provides financial and technical support to Canadian municipal governments toward developing sustainable and environmental friendly communities. With close to 2,000 community members, FCM represents the interests of municipalities on policy and program matters that fall within federal jurisdiction. Members include Canada's largest cities, small urban and rural communities, and 21 provincial and territorial municipal associations.

The PCP (Partners for Climate Protection) program is a joint initiative of the ICLEI and FCM. The Partners for Climate Protection (PCP) program is a network of Canadian municipal governments that have committed to reducing greenhouse gases and acting on climate change. Like the ICLEI framework, the PCP program is based on a five milestones framework to guide municipalities to reduce greenhouse gas emissions. These five milestones are:

1. Creating a greenhouse gas emission inventory and forecast
2. Setting an emissions reductions target
3. Developing a local action plan
4. Implementing the local action plan or set of activities
5. Monitoring progress and reporting results (FCM, 2009, p.4)

Appendix A is the summary of green house gas emission inventory, which have been prepared for the City of Charlottetown as the first step in PCP.

2.3.3 ICSP

In 2005, the Federal Government and the Province of Nova Scotia entered into an agreement where funds equivalent to a portion of the federal excise tax on municipalities would be transferred to municipalities to help fund capital projects related to sustainability (Canada, 2006). One condition of the Gas Tax Agreement is that each municipality prepares an Integrated Community Sustainability Plan (ICSP). The ICSP is defined below for Richmond County, one of the C-Change coastal communities.

“An Integrated Community Sustainability Plan is a long-term plan, developed in consultation with community members, which provides direction for the community to realize sustainability objectives it has for the cultural, economic, environmental and social dimensions of its identity.” (Richmond County, 2009)

2.3.4 MCCAP (Municipal Climate Change Action Plan)

The purpose of the MCCAP (Municipal Climate Change Action Plan) is to create a climate change adaptation plan template to help municipalities to develop their context specific action plans. The two major parts of the MCCAP are (1) adaptation and (2) mitigation planning. Compared with ICLEI’s five-milestone adaptation framework, the emphasis of the MCCAP shifts from mitigation to adaptation. The details of the itemized comparison of ICLEI and the MCCAP adaptation frameworks are shown in Appendix B – “Comparison of ICLEI and MCCAP Adaptation Framework”. The MCCAP’s six-step adaptation frameworks are:

Step 1. Build a Team

Step 2. Impacts and Hazards

Step 3. Affected Locations

Step 4. Facilities, Infrastructure and Service Delivery

Step 5. Pillars

Step 6. Priorities for Action (Fisher, 2011)

2.4 Action Planning

Shapiro mentions that

“Action planning is the process that guides the day-to-day activities of an organization or project. It is the process of planning what needs to be done, when it needs to be done, by whom it needs to be done, and what resources or inputs are needed to do it.” (Shapiro 2001, p. 4)

In the context of climate change, communities need to prepare an action plan in anticipation of an emergency event. Emergencies are “Any incident, whether natural or manmade, that requires responsive action to protect life or property” (Homeland Security, 2008, p.11). In order to respond effectively, an action planning process is developed in five phases, as noted below:

- “1. Understand the situation
2. Establish incident objectives
3. Develop the plan
4. Prepare and disseminate the plan
5. Execute, evaluate, and revise the plan” (Homeland Security, 2012)

In the following sections, two emergency management organizations are presented and their mechanisms for making of an action plan is introduced.

2.4.1 FEMA

The U.S. Federal Emergency Management Agency’s (FEMA) primary task is to support the citizens of the United States. With every resource, they have to protect and assist toward making a plan, response, and recover from natural hazards occurred in the United States. FEMA plays an important role in coordinating the action with local government to ensure each responsible departments, agencies or organizations are get involved effectively. FEMA also provides local government with funding to rebuild the facilities and infrastructures damaged by the major disaster.

The definition for major disaster is “any natural catastrophe (including any hurricane, tornado, storm, high water, wind driven water, tidal wave, tsunami, earthquake, volcanic eruption, landslide, mudslide, snowstorm, or drought), or, regardless of cause, any fire, flood, or explosion, in any part of the United States” (FEMA, 2010, p.82). For each event, action plans are prepared for three periods: before, during and after the severe storm event.

2.4.2 EMO

The Emergency Management Organization (EMO) has a provincial and territorial organization in all provinces and territories in Canada. The EMOs provide information, support action, and allocation resources for responding to the region’s hazards. The EMO defines emergencies as

“situations or the threat of impending situations abnormally affecting property and the health, safety and welfare of the community, which by their nature or magnitude require a coordinated response by a number of agencies.” (EMO, 2011, p.4)

To respond and provide support to Canadian citizens, each province and territory have established their own emergency organization depending on the certain issues and contexts they have.

As a division of Department of Justice, EMO’s responsibilities can be clarified by four areas: (i) emergency programs, (ii) emergency services, (iii) communication/public affairs, and (iv) security and intelligence. Nova Scotia has been involved in improving the 911 emergency telephone system by using the latest technologies and information. As well, a new public alert system is being built as an emergency management tool. Meanwhile, the idea of the EMO business plan has been brought into the implementation of action planning by thinking about the vision, mission and mandate of the EMO. The EMO costs are estimated as follows,

Emergency Management Office			
	2009-2010 Estimate	2009-2010 Forecast	2010-2011 Estimate
Program & Service Area	(\$thousands)	(\$thousands)	(\$thousands)
Gross Departmental Expenses:			
EMO	2,235.80	2, 608.30	2,270.50
Ground Search and Rescue	157.00	196.30	201.00
EMO E911 Reporting System	3,510.20	3,811.50	3,760.20
EMO SARC-NIFD	0.00	403.70	
EMO Disaster Assistance	0.00	4,091.10	280.00
EMO Strategic Services	104.00	81.00	129.00
Total Gross Departmental Expenses	6,007.00	11, 191.90	6,740.70
Additional Information:			
Fees and Other Charges	-68.00	-58.00	-68.00
Ordinary Recoveries	-3,749.20	-10,021.60	-4010.00
TCA Purchase Requirements	0.00	0.00	0.00
Provincial Funded Staff (FTEs)	30.00	31.27	33.00

Figure 2.3 (Source: Nova Scotia, 2011)

2.4.3 Business Planning

As the impacts of more frequent severe storms have cost coastal communities dearly, and have slowed down the pace of development. Considering each community as a firm, local governments are encouraged to operate and manage their emergency services from the concept of business planning.

“A good planning should fulfill long-term strategy or objectives by making a short-term decision. Sometimes, short-term solutions seem hard to realize and cost money, however, how to evaluate and negotiate comprehensively are required to assist people to accept these solutions. A good planning should follow the principles of comprehensive, efficient, inclusive, informative, integrated, logical, and transparent.” (Litman, 2012, p.4)

In the case of climate change in coastal communities, the short-term decision refers to the action planning before the event. Although, it costs resources and time to prepare and teach the public how to response when an event happens, it will improve the overall performance of emergency reaction of the whole society and get benefits in a long-term operation.

2.5 Information Technology and System

With the development of information technology, it changes the way we used to think, work, and solve problems. Information technology is used everywhere in our lives, for example, online banking, online shopping, communication, and social networking. It also provides us with useful tools that we can use to deal different problem in different areas. In the following two sections, two kinds of information system are introduced as an important reference for this research study.

2.5.1 Google Crisis Response

The on-line Google Crisis Response seeks to “make critical information more accessible around natural disasters and humanitarian crises” (Google Crisis Response, 2010). This is a map-based online service that people can use to check critical information anytime when they have access to Internet. The system shows all this information on its own Google Map with different polygons, icons and annotations indicating what kind of problem those areas are currently occurring. Figure 2.4 illustrates the web page and what the kind of information it provides.

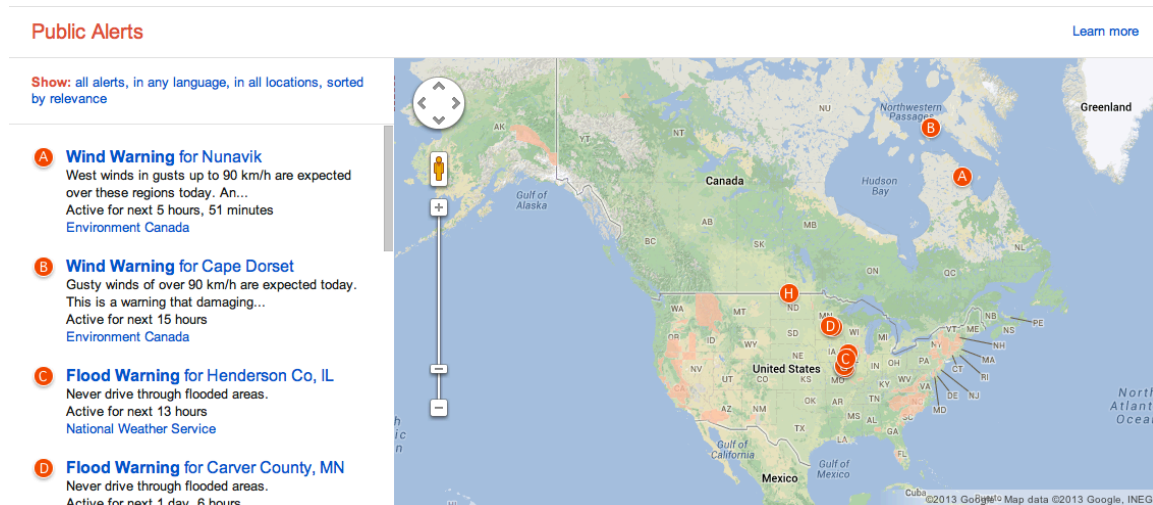


Figure 2.4: Google Crisis Response (Source: <http://google.org/publicalerts>)

2.5.2 Ushahidi

The other crisis information system needs to be mentioned here is Ushahidi. This is a free, open source software system for

“information collection, visualization, and interactive mapping. The software allows individuals and groups to collaborate in creating live multi-media maps for all kinds of projects.” (Lacucci, 2010)

The system reports crisis information through different media like SMS, email, and websites. Most recently, Ushahidi also provides a mobile version of software, which can run on iOS and Android mobile application systems. It receives different sources of input information and collects it. Then by using the platform called SwiftRiver, people can filter and verify real-time information. The Figure 2.5 shows the detailed instruction of how the Ushahidi platform works and what kind of information it provides.

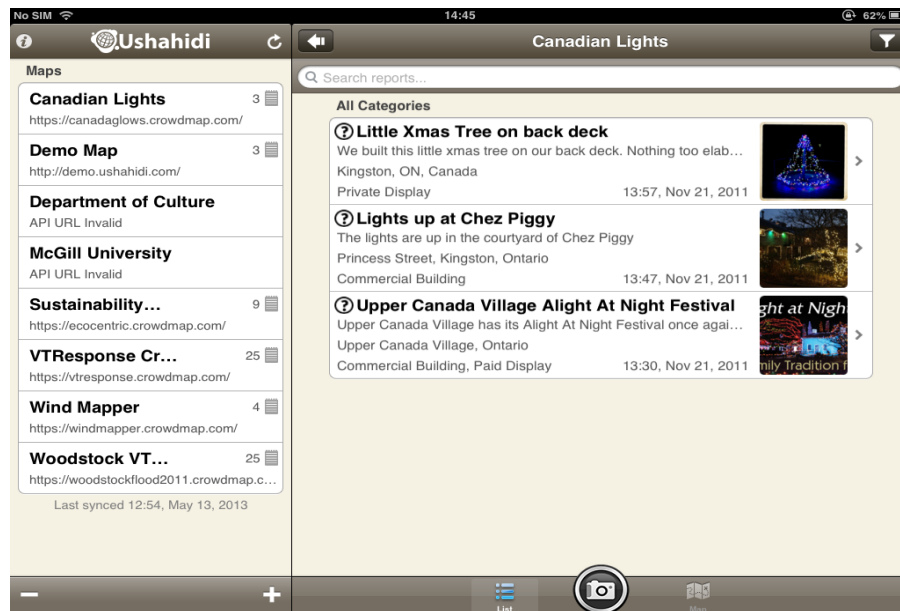


Figure 2.5: Ushahidi Mobile Application (Source: <http://download.ushahidi.com>)

As is noted, on the left side of the window, a table lists all the map areas of interest. On the right side, the master view shows the detailed events or information in each of the areas listed on the left. When users click on the map, shows the location on the map where this event happens, as shown in Figure 2.6 below.

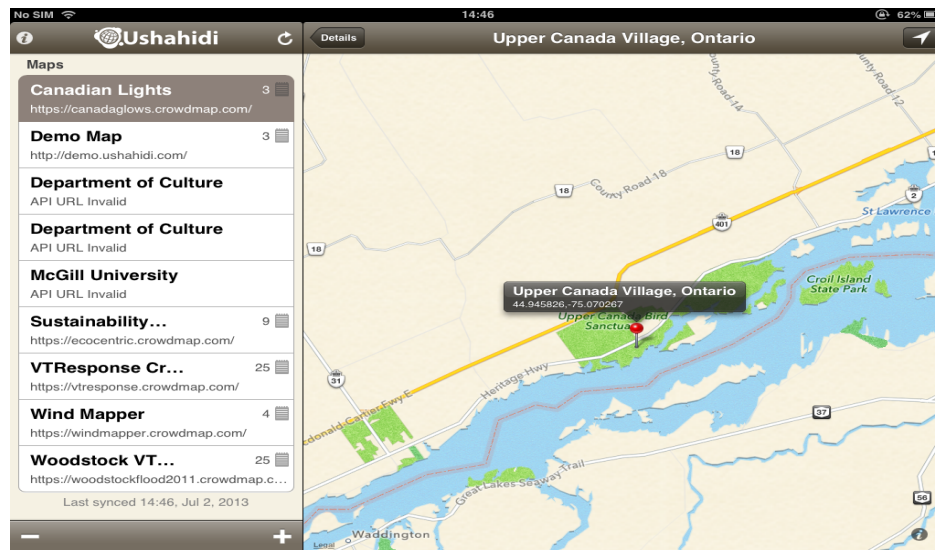


Figure 2.6: Ushahidi Mobile Application (Source: <http://download.ushahidi.com>)

2.6 Community Applications

Since the impacts of climate change has affected so many Canadian coastal communities, local governments and municipalities have involved in the reaction of the impacts by enrolling in the developing adaptation and action plans. In the following sections, two Canadian communities Charlottetown and Isle Madame's applications will be introduced.

2.6.1 Charlottetown's ICSP

The increasing challenges of sustainability of Canadian communities are obvious. The social and economic issues combined with climate change impacts makes the living conditions in some Canadian communities critical. An integrated sustainable plan for these communities is needed to improve the situations for both local governments and citizens in Canada. (Park, Purcell and Purkis, 2009, p.44)

In response, the city of Charlottetown has developed an Integrated Community sustainability plan based on the four dimensional pillars, which are environmental, economic, social and cultural. Each pillar describes the main issues in the community and makes a sustainable plan for the future development. The detailed descriptions and plans are Appendix C - "Charlottetown ICSP Template"

2.6.2 Isle Madame MCCAP

As a jurisdictional part of Richmond County and typical geographic location, Isle Madame has a critical condition due to the increasing extreme weather events and hazards. With only a population of about 3,500 on 20 communities on the island, the information exchange and isolation is the most important issues during the time of emergency. To assist and provide support to these communities, the Nova Scotia government asked for a request of gas tax refund from municipalities within the province to prepare a MCCAP (Municipal Climate Change Action Plan) before the end of 2013. In 2010, Richmond County made a final submission of ICSP report, which is a part of MCCAP program. However, MCCAP focused more on how to make an action instead of

planning. Since many communities in Nova Scotia intent to develop the MCCAP, the town of Yarmouth has made a mentor project for the future development of MCCAP. This mentor is used as a template to start the proposal of MCCAP for the County of Richmond as is shown in Appendix D -“ Richmond County MCCAP Template”

2.7 Summary

In this chapter, ICSP introduced the concept of community profiles corresponding to four pillars that provide a concise description of community decisions. The ICLEI, FCM and MCCAP adaptation frameworks have much in common and provide a consistent structure for subsequent analysis of adaptive program contents. Adaptation problem seems more clear and easy to implement. The adaptation frameworks provide communities with basic process of adaptation and consist in developing a more detailed adaptation plan. In the next section chapter 3, based on the review of ICLEI and MCCAP’s adaptation framework, an adaptation framework is developed by including the concepts of community profile, threats, and action planning.

3 Methodology

The purpose of this thesis is to develop an integrated adaptation framework, which can utilize the information that the community provides to make a clear description of the coastal community's changing environmental situation and implement adaptation actions. Then, based on the results of adaptation, implement an action-base program to assist the community members in order to improve preparation and emergency response. In this chapter, a static and dynamic system is designed using information technology for both adaptation and action. This includes the development of a social mapping and community member electronic alert status system within a community action-based website to showcase both the municipality's MCCAP and the community status alert system.

The methodology described below includes sections that describe: (1) the research process model adaptation and action framework; (2) the system analysis and design; (3) the data and information process; and (4) aspects of implementation and the research process.

3.1 Process Modeling

The fundamental idea of developing the adaptation framework is based on the concept of problem identification, problem analysis and problem solving. A well-developed framework gives communities a starting point to get engage in the coastal climate change adaptation issue. In terms of system science, the framework should be able to be easily built and updated when there are new requirements from municipalities and communities. Many Canadian coastal communities have experienced severe impacts related to their changing environment, Global and Canadian organizations, e.g., ICLEI and FCM have attempted to provide broad strategies and recommendations for these problems. For example, PCP (Partners for Climate Protection) as a part of FCM has developed a five-milestone framework for the reduction of GHG and adaptation planning that is based on ICLEI's five-milestone framework for adaptation. Similarly, the MCCAP has been promoted by FCM and is being implemented by the provincial government of Nova Scotia. Adaptation elements in the MCCAP are used in this research toward developing

and applying a more extensive for applied and comprehensive use. The process used to develop the adaptation plan and related to the MCCAP process (Fisher, 2011) is described in the four-step process of Figure 3.1 below.



Figure 3.1: Climate Change Adaptation Process

For each of these steps in the adaptation process, detailed process elements are defined to simplify the complexity of climate change problems from a system point of view. The principles of making this adaptation framework are to integrate current information and profiled community research results in order to respond to the impacts of climate change, improve the problem solving and decision making process, and transform the framework into an action-oriented applied system, which people can easily understand and access at the community level.

3.1.1 Community Profile

The first part of the adaptation framework is the community's profile. Community data collection includes basic background information, historical research results, and current status about the status of the coasts, the people, and their resources. Since some communities have already done local research and work related to the reduction of the negative impacts of the changing environment, this relevant information can be used as part of the community profile elements (Hartt 2011, Pakdel 2011, Boudreau and Theriault 2012).

The elements of the community profile should be highly integrated and easy to obtain. To achieve this, the four-dimensional pillar for community sustainability is taken from the ICSP (Richmond Integrated Sustainability Plan, 2010) and inherited as the main structure of the profile framework applied in this research. These four pillars are the: (1) Environmental, (2) Economic, (3) Social and (4) Cultural community pillars. In this research, the community profile is described in further detail below. This information emphasizes location-based information and is applied to the community application of Richmond County presented in the following Chapter 4 – System Analysis and Implementation.

3.1.1.1 Environmental Pillar

The community environmental profile presents the natural resources of the community, its geographic location and information such as the coastal hydrology. The elements of the environmental profile are summarized in Figure 3.2 below.

Recreational	Resources	Developed
• Parks • Trails • Beaches	• Forest • Wetland • Marine • Wilderness	• Recreational • Commercial • Public

Figure 3.2: Community Profile-Elements of the Environmental Pillar

3.1.1.2 Economic Pillar

The community economic profile presents the commercial activity and industrial production activity. Local job opportunities are provided and the significant community contribution to economic growth in the entire region. In many coastal communities, substantial revenues come from tourism related business. Each year when the tourist

season comes, relevant events take place as the main activity in community's daily life. Public works and infrastructure are important elements in the economic pillar. These elements are responsible for keeping the community safe, and clean, and reduce negative environmental impacts. The elements of the community economic profile are summarized in Figure 3.3 below.

Commercial	Public Works	Infrastructures
• Retail • Tourism • Agriculture • Manufacturing	• Police Station • Fire Station • Transportation • Government Building • Waste	• Wharve • Fishing Hut • Ports and Harbour

Figure 3.3: Community Profile-Elements of the Economic Pillar

3.1.1.3 Social Pillar

The Social pillar indicates the density of the community population in different areas, and is helpful in describing the community's future development and civil status. The social profile includes the public health service where community members can receive health services. The levels of educational institutions are also listed as an important social asset. The elements of the community social pillar are summarized in Figure 3.4 below.

Population	Public Health	Education
• Environmental Density • Commercial Density • Residential Density	• Hospitals • Clinics • Health Care Centre	• School • University • Kindergarten

Figure 3.4: Community Profile-Elements of the Social Pillar

3.1.1.4 Cultural Pillar

Communities' special and unique cultural backgrounds are under risk from coastal environmental changes. It is part of the municipality's responsibilities to represent and protect this cultural heritage and enriches it for the future generations. Cultural considerations acknowledge faith-based backgrounds and country or heritage of origin along with the community's associated events and activities. The community is the perfect place to provide each cultural group to exchange information and share time-honored experiences with each other. Knowing the norms of the cultural community also permits decision makers' opportunities to participate and learn more about feasible adaptive capacity to the changing community coastal environment. Galleries and museums are noted as important landmarks and physical cultural assets in presenting the special and unique aspect of the cultural community. The elements of the community cultural pillar are summarized in Figure 3.5 below.

Places	Religious Sites	Communication
• Museum • Historical Site • Library • Cultural Centre	• Church • Cemetery	• TV Cable • Tower • Radio Station

Figure 3.5: Community Profile-Elements of the Cultural Pillar

3.1.2 Threats and Vulnerabilities

Once communities' assets have been profiled, the next step in the adaptation and action framework is to identify the threats to the community and how the community is vulnerable especially to the natural forces from more frequent severe coastal storms, storm surge, and sea level rise. If we consider the coastal community as a system, threats of climate change are exogenous events that communities cannot control. Further, community threats and vulnerabilities make the potential adaptation situation different from each community. However, if communities can clearly identify their threats and

vulnerabilities and how these affect different aspects of the community profile, they can more effectively use their limited resources to prepare and reduce negative impacts. In order to understand the threats and vulnerabilities, there are three things coastal communities must consider. These are: (1) the type of threat, (2) when the threat may occur, and (3) where it will affect the community. These steps of threats and vulnerability identification, and their impacts are presented in the following subsections.

3.1.2.1 Types of Threats

(1) *Sea Level Rise*: rising seas erode communities' coastlines and damage properties that are not sufficiently set back from the shore. This includes, importantly, community infrastructure (e.g., water and sewage systems) that were historically constructed in the areas of foreshore to facilitate their operations. Sea level rise is a long-term mainly one directional phenomenon that occurs in small positive and annual increments (e.g., measured as millimeters of rise annually). Together with land subsidence experienced in many areas, the local sea level rise event may be greater than forecast for whole coastal regions. More frequent and more severe seasonal flooding of shorefront properties or coastal roads are typical impacts resulting from rising seas.

(2) *Storm Surge*: When a storm hits coastal communities, everything on the path would be affected according to storm severity, e.g., increasing hurricane category. Severe storms associated with floods or hurricanes may cause massive destruction to the community. Alternatively, coastal storms with seasonal high winds, rapid precipitation, and high tides can cause maximum high water levels despite being identified as a mere tropical depression according to how the local community is "hit" by the storm.

(3) *Ice Melting*: As a direct result of climate change, warming temperature decreases the volume of the ice sheet and glacier. According to scientific research, the rate of melting keeps increasing and the ecosystem patterns changed dramatically.

(4) *Erosion*: Wind, precipitation aid in shifting coasts and beaches– often a result of poor land use from agricultural run-offs, clear cutting lumbering policies, and ineffective shorefront armoring.

3.1.2.2 *Timing of Threats*

The timing of severe storm events that result in storm surge and flood damage are regularly most important as threats to the vulnerable coastal community. The impacts of these events are exasperated by steadily rising seas. Threat timing are described simply as follows:

(1) *Seasonal*: For most coastal communities in Canada and the Caribbean regions, severe weather events happen seasonally primarily because of the regular interval movement of ocean current. The time period between seasonal events can be months or even years. Communities can consider these events as predictable and the only thing uncertain is how severe the events may be. For example, based on the forecast report from North Carolina State University, upwards of 13 hurricanes are expected to occur on the eastern seaboard of the United States (Xie, Hammerling, Liu and Fuentes, 2013) According to such historical weather report, communities can get an approximate idea of when the average level of seasonal storm events that can be expected to occur.

(2) *Occasional*: Severe weather event may happen without seasonal or other signs or notifications and nevertheless cause large scale of damage, e.g., in Atlantic Canada, winter storms (November to February), not identified as hurricanes or tropical depressions, may result from southerly, westerly, and/or easterly fronts colliding. Most often when such events may occur, communities do not have enough time to prepare or react for weather that may include storm, rain, ice, and high winds and tides. A well-prepared emergency response plan can help communities to reduce vulnerabilities and incurring additional damages costs.

3.1.2.3 Location of Threats

The actual location of severe storm events relative to the vulnerable areas of the community is important to consider. These locations are discussed below.

(1) *Vulnerable Areas*: The places under threats or have potential risk can be defined as vulnerable areas. Communities often declare information about areas at risk of damage is in these designated vulnerable areas that may be precariously close to shore, or on unstable land areas. Prepared information such as presented in Isle Madame Research Project (Boudreau and Theriault, 2012) provides communities with a ranked collection of typically known information about designated vulnerable areas. Communities also use this information to evaluate and estimate the cost of storm impacts and strategies for adaptation to reduce vulnerabilities.

(2) *Vulnerable Places*: Places, i.e., residential and commercial buildings, infrastructure, bridges, and other sites, under threat are singular spot locations with particular vulnerability. Updating geographic information of these spot locations, communities can monitor how they change and act based on the adaptation priority.

Figure 3.6 presents a summary of the structured threats to coastal communities,

Threats Type	Threats Timing	Threats Location
• Sea Level Rise • Storm Surge • Ice Melting	• Seasonal • Occasional	• Areas • Spots

Figure 3.6: Community Threats Framework

3.1.3 Adaptation Team

The next step in the MCCAP process is the assignment of the adaptation action team. The involvement of members in the community can effectively reduce the adaptation pressure of the governing municipality. In the following sections, the participants in the development and application of the climate change adaptation and action plan are declared, and then analyzed with regard to the functions of each participant and their assigned roles and responsibilities in delivering response and recovery from severe storm events.

3.1.3.1 Team Structure

From government to community, the adaptation and action team is built around four different levels, namely: (1) the federal government, (2) the provincial government, (3) the municipal or local government, and (4) the local community and its members. These four main participating groups play important roles in the process of community change adaptation.

From the decision-making perspective, the federal government is on the top level of making policies or regulations, while the provincial and municipal governments follow. Local communities appear at the lowest level of direct decision making. Figure 3.7 below shows the general hierarchy of the adaptation team.

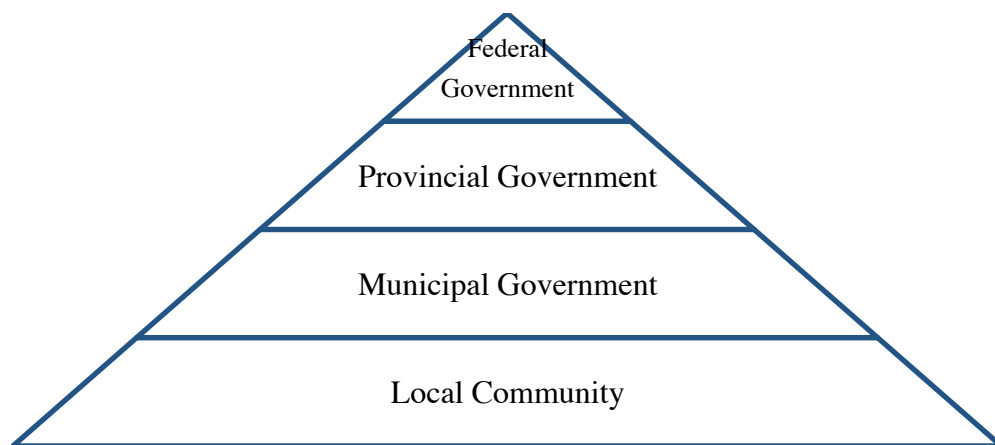


Figure 3.7: Community Adaptation Team Hierarchy

3.1.3.2 Team Analysis

The following paragraphs describe the roles and responsibilities of the team members presented in Figure 3.7.

(Level 1) Federal government: As the highest level of decision-making, the federal government can set national policies or legislation. Normally, each new policy or legislation takes a long time to discuss and publish, which cannot immediately meet the provincial or local governments' needs. However, in some critical situations such as natural disasters, the federal government has the power to deploy special resources that are not used frequently in normal conditions. The federal government does not only make policies, but also releases funding and resources to provincial and local governments to develop the local economy, improve living conditions, and provide better public service.

(Level 2) Provincial government: Provincial governments also have the authority of legislation, however, they are more responsible for public service, the health system, and provide assistance to local governments (municipalities). They can make policies directly to the issues municipalities currently face, and provide valuable advice and suggestions. They may also provide funding and resources to municipalities to deal with certain issues.

(Level 3) Municipal government: The function of the municipal government is to manage facilities and infrastructures, provide local public services, and make development planning for local communities. As part of public service, departments such as fire stations, police stations, or paramedics are common municipality resources that may be deployed in emergency situations. Another important function of municipalities is local facilities and infrastructures management. Municipalities are in charge of civil design, new constructions, and local road repair. With regard to the impacts of climate change, municipal projects should consider environmental sustainability as an important component of their decision-making.

(Level 4) Local community: In the last level of adaptation team structure, the local community can be defined as groups of businesses, local professionals, universities, industries, churches, councils, committees, and non-governmental organizations (NGOs).

Each of these groups plays an important role in local development. These groups can be categorized into two parts: profit-motivated groups, and non-profit groups. Taxes paid by profit groups and individual taxpayers' contribute the majority portion of the local government's operating income. Non-profit groups, e.g., NGOs, churches and volunteers groups are key participants for local social and cultural activities. These groups work closely with community members, and acquire first hand information about what people in the community need and how to help them.

3.1.3.3 Roles and Responsibilities

As discussed above, the federal government and provincial governments have similar functions in policy-making and legislation at the national and regional (provincial) levels. They support local municipal governments with relative policies and funding. In order to adapt to climate change locally, municipal governments, as managers of the local community, should understand current climate change issues and use their resources appropriately to make both short-term and long-term adaptation plans. In emergencies, municipal governments also have the responsibility to response and deploy resources effectively to reduce the impacts of climate change.

Climate change is a complex problem, which requires multiple groups' involvement. At the last level of adaptation team, local communities have large numbers of people who are under the threats of climate change. However, at the local community level, local information arrives faster than municipal governments and community members can make a response quickly. If these local groups can be well organized and trained, they can effectively reduce the adaptation pressure of municipal governments. This research focuses on how to utilize local communities as main participants in the response of climate change impacts and how to build up the communication between municipality and local communities to cooperate with each other in the situation of emergency. This perspective turns the structure of the adaptation team (Figure 3.7) upside down and realizes an alternative view as follows in Figure 3.8.

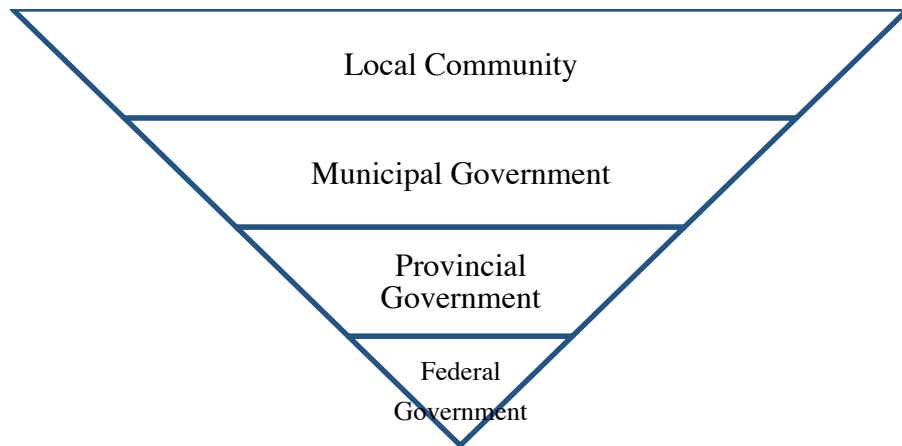


Figure 3.8: Reoriented Adaptation Team Significance

3.1.4 Action Planning

As the next step in the MCCAP process, the timing of action plans can be simply defined as three periods: (1) before, (2) during and (3) after the emergency event (Figure 3.9). Before an event happens, communities should focus more on how to prepare the resources and teach people how to respond when an emergency occurs. The concept of making an action plan for the second period is to consider all the possibility that what will cause the major failure during the severe weather event. In this period, immediate actions are required to deal with any kind of emergency. In the last period, actions are based on how to recover from the event.

To start action planning, communities need to prioritize the essential problems. Some action aspects can be defined as typical issues, which communities have experienced in the past. By the time communities have more knowledge about the changing coastal environment from historical review of the threats and vulnerabilities (e.g., Boudreau and Theriault 2012), the action aspects can be updated to deal with the latest issues. The following section summarizes the basic action aspects needed by coastal communities.

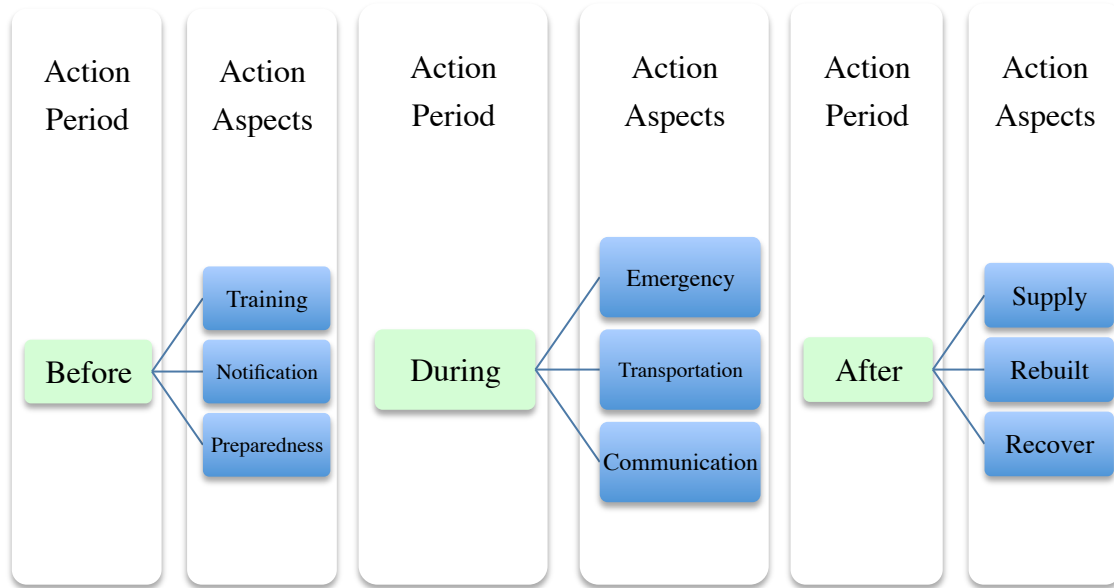


Figure 3.9: Action Planning Aspects: Before, During, and After

According to each community's changing environmental conditions and adaptation needs, detailed action plans can be prepared by filling out the action aspects. Examples of action plans are introduced in the sections below.

3.1.4.1 Before

First, in the action aspect of preparing and training, local communities learn and understand how their changing coastal environments affect their living and social conditions. Municipal governments have the responsibility to pass information to local community members, such as expected impacts, risk and vulnerability, emergency preparedness and response, etc. By the time local communities have enough knowledge of these information and skills, they can also help each other. This is an advantage of small, local communities who participate directly in the action plan related issues as adaptation team members. Another action and preparatory activity is the storm event “table top” or storm event “mock-up” exercise in which local municipal governments actually practice the critical decision events of the emergency situation in order to be better prepared for real response. By doing this, municipal governments and the adaptation team would have a better understanding about their action capacity and preparedness capability.

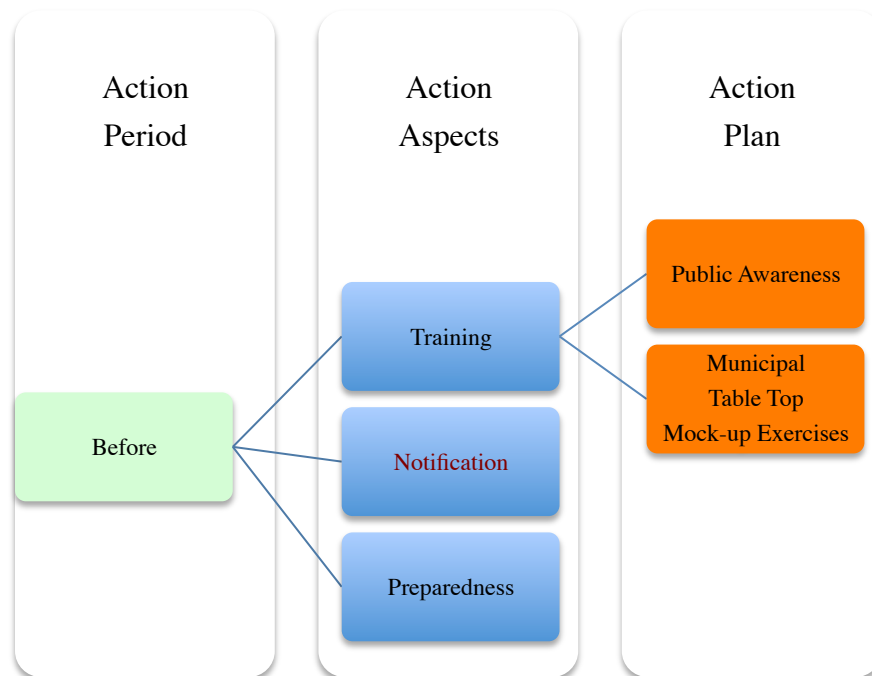


Figure 3.10: Before Events: Training Aspect

In the notification aspects, monitoring and warning system are two actions municipal governments need to consider. Climate change has changed the patterns of global weather systems and severe weather events happen more frequently in some regions. According to the discussion above, the timing of climate change threats, weather events happening more often as seasonal events but at more severe levels, while other severe storm events occur abnormally, with more damage and destruction. The municipal government and the adaptation team needs to monitor weather forecasts by checking the real time radar information or using some other kind of technology to keep the local communities updated. Such a community warning system develops warning levels in order to communicate emergency situations to communities. The level of this system depends of how likely and severe an event would be. Normally is defined as three levels, which are attention (low), action (medium) and evacuation (high). Depending on the level of the system, municipal governments have to prepare certain immediate action plans accordingly.

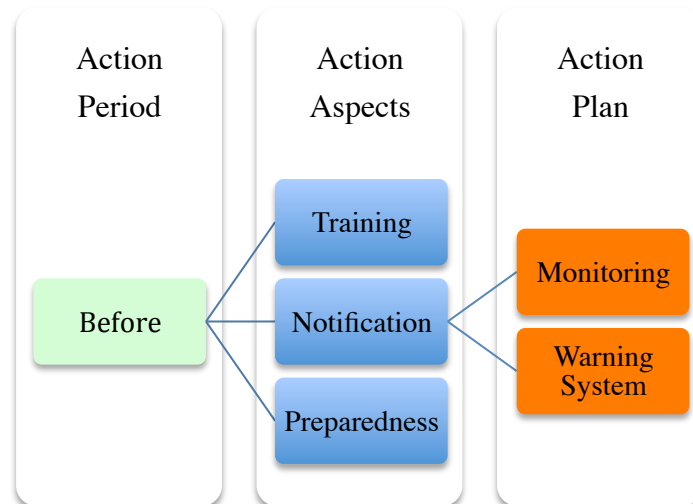


Figure 3.11: Before Events: Notification Aspect

The last aspect of the “before” period is preparedness. Sometimes it seems hard for municipal government and local communities to decide what should be prepared before an emergency event happens, because the complexity of climate changes impacts and the difficulty of predicting how the event would affect the community. To simplify this problem and easy to start, two important actions municipal government need to consider, vulnerable areas and backup resources. Vulnerable areas are the places or spots defined to be risky during an event. These areas should be defined when municipal governments identify the area of climate change threats in the above 3.1.2.3 section. The preparedness can start with these vulnerable areas by building defenses or evacuate people in these areas. In the situation of emergency, people may lose essential resources such as water, food, accommodation, and power. If someone is injured, medicine and first-aid are required. These are all the resources municipal governments need to prepare before an event happens. It is also important for the people who live in the community to prepare themselves in case of an emergency, because municipal governments may not provide these things immediately. What to prepare and how to prepare is the course municipal government need to teach local communities in the above training action aspect.

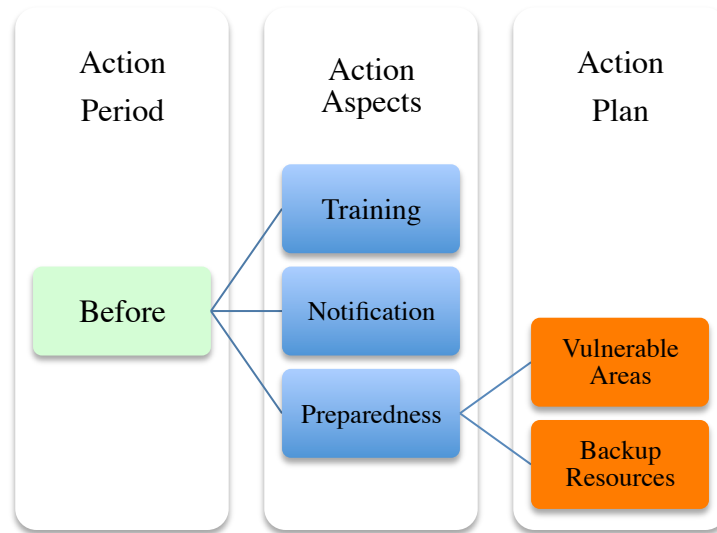


Figure 3.12: Before Events: Preparedness Aspect

3.1.4.2 During

The more municipal government and local communities are prepared before an event, the less the need to be concerned about the welfare of community members during the emergency event. However, there are still some essential aspects municipal governments need to plan during an event, which are emergency reaction, communication and transportation.

The most common thing happens during a severe weather event is emergency requests from the people who are in trouble. Department like fire stations and police stations are responsible for any kind of emergency request send from people and the priority of rescue people should higher than any other request. Paramedics should stand by in case someone gets injured during the emergency. Another event happens frequently during an emergency is fire. From previous experience of severe weather events happen to coastal communities, power plant and gas station are the places cause fire. At that time, fire department has responsibilities for both the safety of people and control the extension of fire. People who live in those areas should be noticed to leave before situation becomes worse. The last emergency situation municipal governments need to consider is pollution, if there is any nuclear plant or chemical factory in their regions. These places need to be

marked as dangers areas where local communities should be informed before the event. Different action plan scenarios have to be made in order to reduce any potential risk cause by the failure of controlling the pollution.

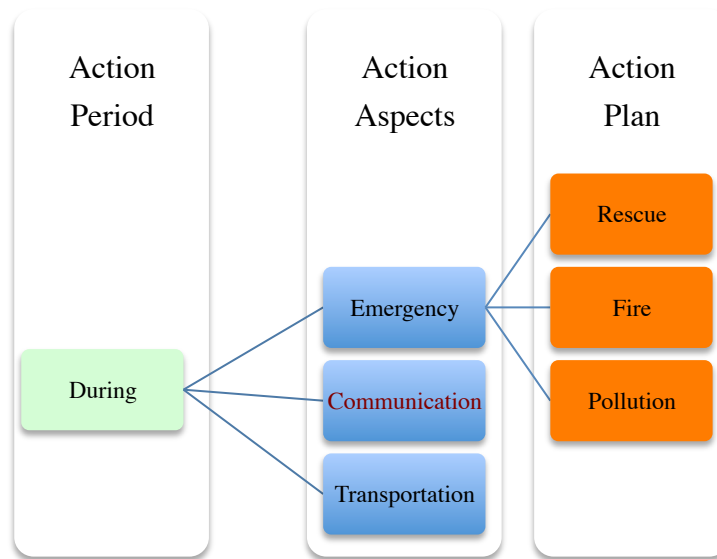


Figure 3.13: During Events: Emergency Aspect

As the second action aspect during an emergency event, municipal government need to build up ways to communicate with local communities for both alert and reaction use. By receiving the latest alert information from municipal government, people in the communities would react appropriately. Municipal government can make an announcement of this information by using TV channels, radio broadcast, and Internet. Another action plan for communication is the emergency requests from local communities when someone is in trouble. The department like fire stations, police stations, and paramedic fleets need to response immediately to the requests send from local communities and cooperate with each other to deal with difficulty situations. The most common way for municipal government to receive emergency requests from local communities is by phone calls. It is important for municipal government to get the information about what the problem is and where does it happen, so the emergency department can quickly make an action plan accordingly. As is discussed above, local communities as an important participant of climate change adaptation team can also provide help to the people who live around them if the situation is not critical. In order

for the people to communicate with each other in local communities, the information of the people who want to join the contact to list to provide help to someone else need to be collected and post to each household.

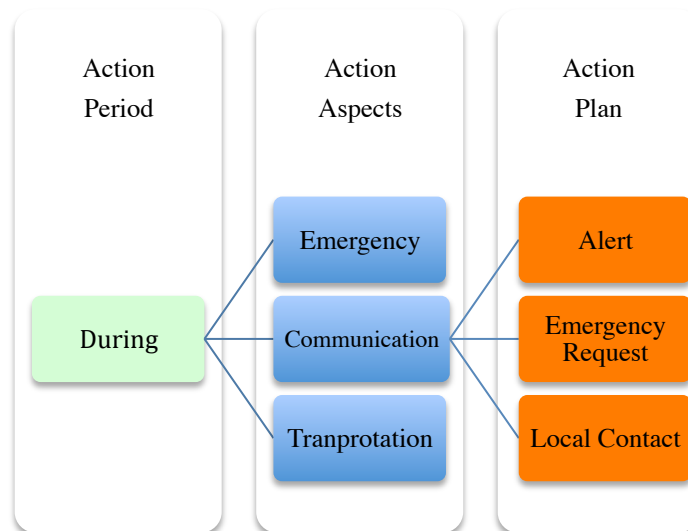


Figure 3.14: During Events: Communication Aspect

The last aspect during an emergency event is the condition of transportation. During a severe weather event, facilities and infrastructure can be damaged deal to the cut-off communication and transportation, which makes people who live in those areas cannot get support on time. To avoid people moving into those areas, local communities need to know where to be considered as dangers areas. Municipal government can either provide this information from the requests report send to emergency department or inform local communities with the areas defined as vulnerable. At the same time, municipal government need to indicate some safety routes to emergency centers where people can go for help. If someone need to move out from the dangers area, but without transportation, municipal government needs to send emergency fleet to those places to help people get out of there.

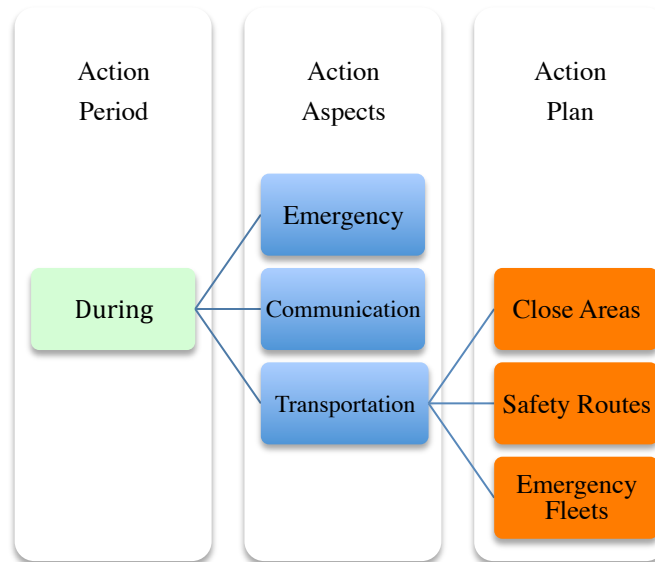


Figure 3.15: During Events: Transportation Aspect

3.1.4.3 After

Since communities have experienced the hard period during an emergency event, life is still not back to normal. People who lose their properties and assets need to get supplies for temporary use, facilities and infrastructure damaged should be rebuilt and local economies take time to recover. All these works should be well planned and cooperated. Detailed action plans for after emergency event are introduced as follows. Water and food are two important things municipal government needs to prepare as supplement, which is similar like the preparedness action before an event. They should be stored in some places where impacts may not affect. The quantity depends on the size of community and how many population they have. Sometimes, a severe weather event can cause massive destruction like house destroy and infrastructure damage. In this kind of situation, municipal government need to have an action plan for providing accommodations to the people who lose their houses. The places for accommodation can be some public areas like schools, churches or arenas.

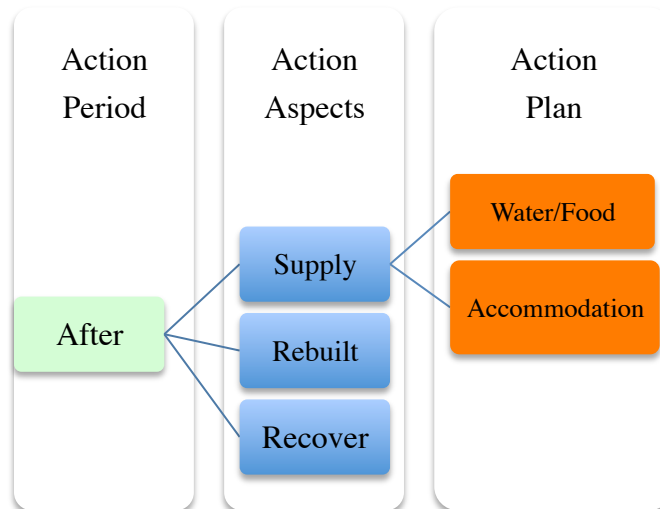


Figure 3.16: After Events: Supply Aspect

In the Rebuild aspect after an event, action plans need to be made for both infrastructure and buildings. Destructions and damages to infrastructures would cause power and water supply outage or transportation block. If municipal government can rebuild these important elements effectively, local communities would get back to normal life faster. The other part of rebuild after an event is ensuring the safety and status of buildings. Action plans need to be made for both municipal government buildings and local community buildings. The damaged buildings need to be detected by construction specialist before people can work and live in them. If specialist confirms it as dangers, then re-construction should start until it is proven to be safe.

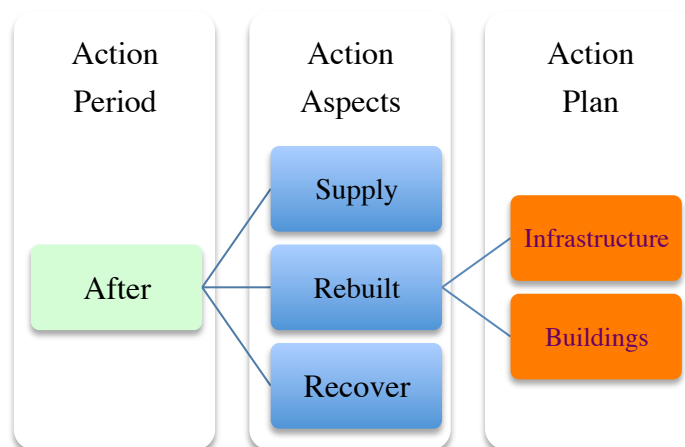


Figure 3.17: After Events: Rebuilt Aspect

The last action aspect after an emergency event is Recover. According to different levels of destructions, the recover period can be months or even years. Federal government and provincial government may also get involve in as provider and technique support to help municipal governments and local communities to get back to normal. There are two important recover actions municipal governments need to consider, which are public service and local economy.

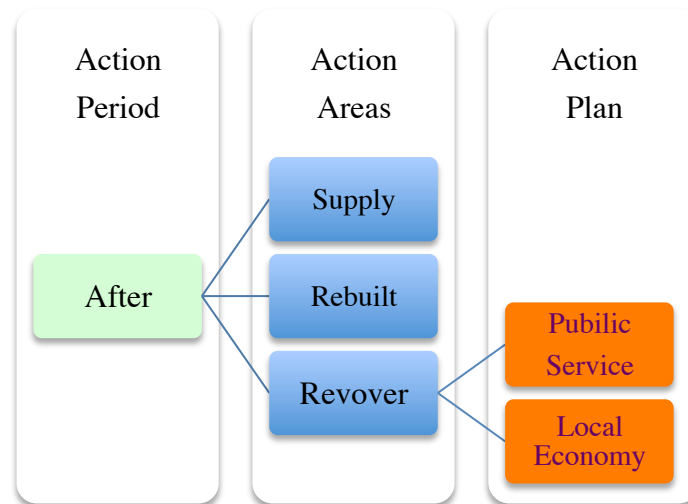


Figure 3.18: After Events: Recover Aspect

3.2 System Analysis

An effective adaptation plan requires both municipal government and local community's participation. Traditionally, the communication between municipal government and local community is described in Figure 3.19 below. Municipal government typically is flooded with information and often do not have enough capacity to output and respond to all requirements. An more appropriate means of exchanging information within the community is to use the local social network of community members as in Figure 3.20, in which local community members can receive information from both municipal government and other community members as input and then respond with certain localized, relevant, timely, and neighborly actions. In order to achieve this goal, two kinds of technologies are discussed as key action items in this research:

- (1) the use of the extensive Internet within the community; and

- (2) the development of a mobile-based system for the communication and response of community members to local emergency situations.

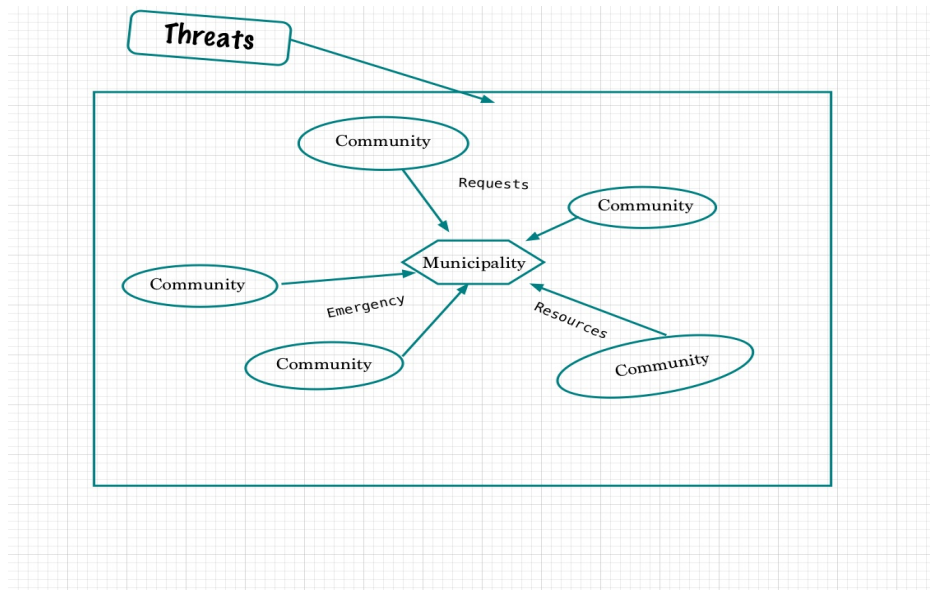


Figure 3.19: Current Centralized Local Adaptation Situation

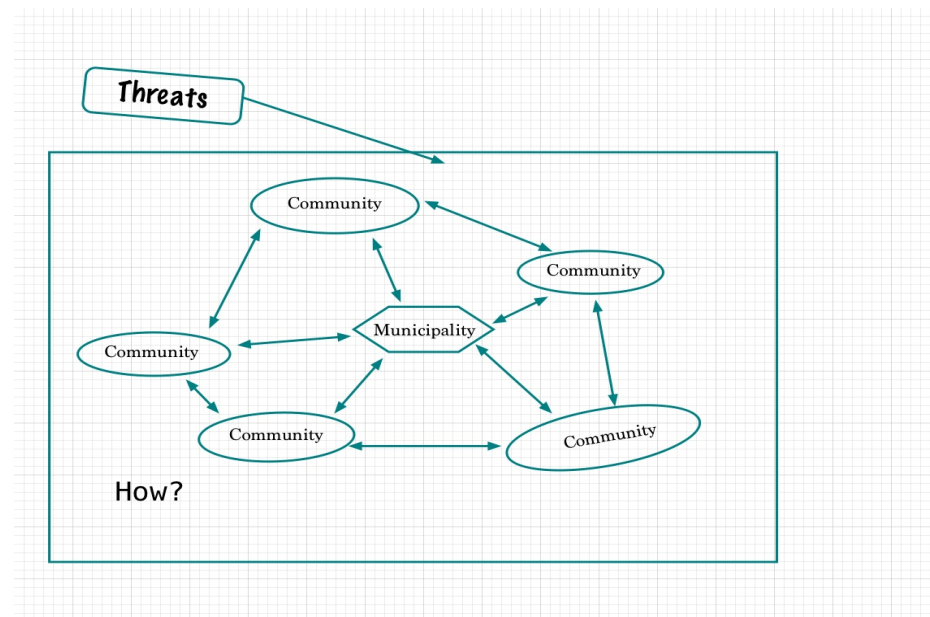


Figure 3.20: Improved Local Community Adaptation and Response Situation

(1) *Internet*: For multiple ways of communication, the expansive availability and use of the Internet is an appropriate implementation. Social networking tools like Facebook, Twitter and Flickr are very popular and many community members are already familiar

with using these to share and receive information from their families and friends. Thus, it is fundamental to this research that the Internet, via social networking, enables more effective sharing and communication among community members to share and receive information in the adaptation process to the changing coastal environment. At this point, municipal governments and local communities can communicate with each other in the way shown in Figure 3.20.

(2) *Mobile Technology*: With the development of mobile technologies like 3G/4G, smart phone devices and tablets are utilized as the most popular information carrier that people can communicate with each other at any time. These technologies allow people to get access to the Internet much more easier and convenient than the traditional ways like DSL, Cable or Wi-Fi. These technologies also set people free from the certain locations like home, office, school where they can connect to Internet. The other important feature of mobile technology is that it could locate users geographic information.

Based on the above analysis and discussion, in order to take the both advantages of these two technologies into climate change adaptation, certain tools and platforms need to be selected. The detailed information tools and technologies used in this research to develop this system is presented as follows,

Joomla: As a popular open source website development tool, Joomla is widely used in website development and management. Before start developing a website, a domain on a server is required to allow developers upload and download website contents. The domain name used in this research is from adopted from C-Change project website, which is www.coastalchange.ca/richmondcounty. All the articles update, news post, and template change can be done through Joomla's control panel, which is powerful and user friendly.

Xcode: Right now, there are three main mobile operating systems in the market, which are iOS (Apple), Android (Google), Windows (Microsoft). Based on the stable ecosystem and high volume of users, the system in this research for climate change

emergency response would be implemented upon the iOS system for the devices of iPad and iPhone by using Xcode. Xcode is developing tool, which is used to develop OS X and iOS applications. The language used in Xcode to develop mobile applications is called Objective-C, which is an Object Oriented programming language designed specially for iOS devices (iPhone, iPod touch, and iPad). The code used to develop this application is shown in Appendix E.

Parse: In order to develop a mobile application, which allows users to communication with each other and receive information, a well-developed backend service is required to fulfill this. Traditionally, organizations or companies need to buy physical machines and hire technicians to maintain this machines in order to provide continuous data exchange services. In this way, it costs a lot of money to build this infrastructure and keep it functioning. Now, with the development of cloud service, some IT companies like Google, Amazon are providing different kinds of cloud services, which allows organizations and companies to spend less money and get professional backend services. Parse is a specific cloud service company, which supports mobile applications. It provides professional services like data storage, data management, and SDKs. Developers can simply connect their mobile applications to Parse's servers to receive and update data in the database by using its powerful APIs.

3.3 Information Process

The community profile information, actually presents what kinds of properties and assets communities have in their region. The information can be updated by the time when communities add or remove some elements, but the way it categorized should follow the four-dimensional pillars, which are environment, economy, social and culture. The process of transforming the profile data into information is based on three steps: what is the name of the property, what kind of function does it have and where is it.

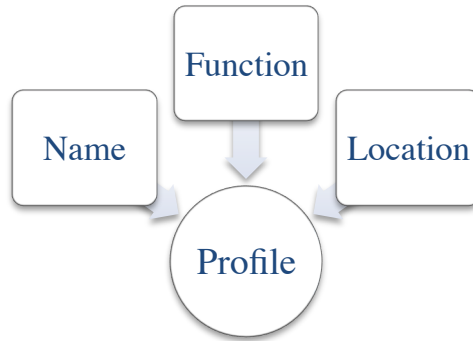


Figure 3.21: Community Member Profile Information

To process threats data into information, communities have to define the type of threat, analyze when it will happen and where are the vulnerable areas. The only information does not change usually in communities' threats is the type. For both time and locations, communities can only use past experience or new analysis method to estimate and evaluate. Associated with community profile information, communities can clear identify which profile elements are under threats and where they are.

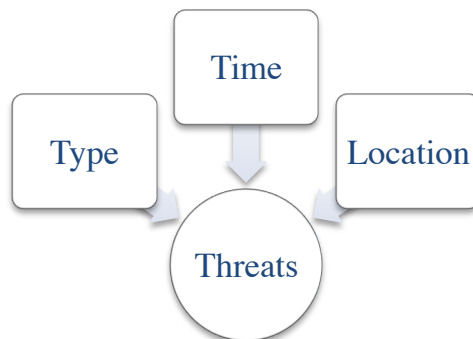


Figure 3.22: Threats Information

The third part of processing adaptation information is community's adaptation team. For each participant in the adaptation team, communities have to know what kinds of resources can they provide, what are they responsibilities and where are they located. The adaptation teams' resources information is about what kinds of help or support they can provide. To have this information, communities would understand who has the adaptation resources and make a request. The other important information for communities to have is the adaptation teams' responsibilities. If each participant knows clearly what they are responsible and how to response, it would help communities to solve problems

effectively and appropriately. The same as other adaptation information, it is helpful for communities to have the location information about where the adaptation teams are, so they would find out which areas need more coverage in the case of emergency event.

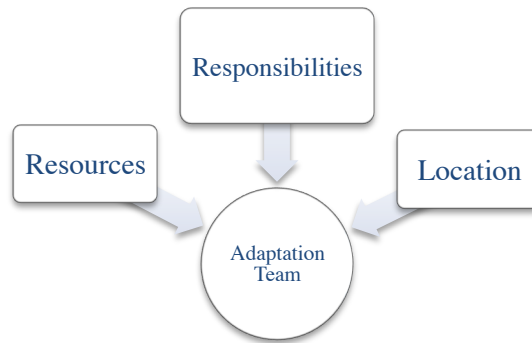


Figure 3.23: Team Information

At last, for each period of climate change action planning, the action target, who takes the action and where to act is the information each adaptation team need to collect. The target of an action presents communities adaptation needs during the three periods of time. For example, before an emergency event, local communities are the targets, which municipal governments have to teach and assist. During an event, municipal governments' action would focuses more on emergency response and vulnerable areas. To collect the action participants' information, both municipal governments and local communities would understand who are involved and where else need to cover. Then, locations information of action target helps each action participant to make a decision of where they can provide help.

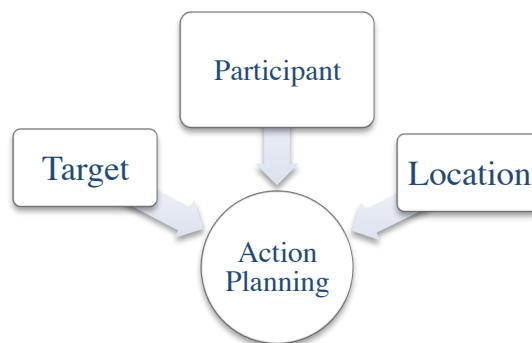


Figure 3.24: Action Planning Information

In summary, this chapter started with the development of climate change adaptation framework based on the review of Municipal Climate Change Action Plan (Fisher, 2011) and Local Government for Sustainability (ICLEI, 2012) adaptation models. The processes of adaptation modeling in this research are Community Profile, Threats and Vulnerabilities, Adaptation Team, and Action Planning. The last section in this chapter converted adaptation needs into a system requirement for the preparedness of implementation.

According to the framework development and system analysis above, the next chapter implements the current information technology to meet all the system's requirements defined above. The steps of the implementation are based on the workflow of application development, which follows the process of design (defining application functions according to the system's requirements), validity (using program language to implement the functions), and verification (testing the application's functionality).

4 Implementation

This chapter presents the results of the system analysis and technologies introduced above in Chapter 3 to implement the adaptation action plan for the C-Change community of Richmond County, Cape Breton, Nova Scotia, Canada. This includes the step-by-step development of the mobile application for community emergency communication, and the adaptation action plan website. These steps, as noted below range from (1) community adaptation website development, (2) community adaptation mobile application design and functional analysis, including mapping, user interface, control, and programming; to (3) model validation, testing, and verification of mobile application.

4.1 Community Adaptation Website Development

As is shown in Figure 4.18 the adaptation mobile application's Home View, there are three different colors square button on the right side on the Home View are the links that lead users to a website, which is design as an alternative way to provide community members adaptation information. These three links are Local News, Events and Policy under the main menu of this website Community Life's tab. Joomla is the developing environment for this website's design, manage and maintenance. As introduces in 3.2 System Analysis section, Joomla has been proven a good tool in website development. It is easy to use and manage. Figure 4.1 shows the Control Panel of Joomla where developers can use to develop and update a website. As we can see, there are many specific section developers can use to manage different components.

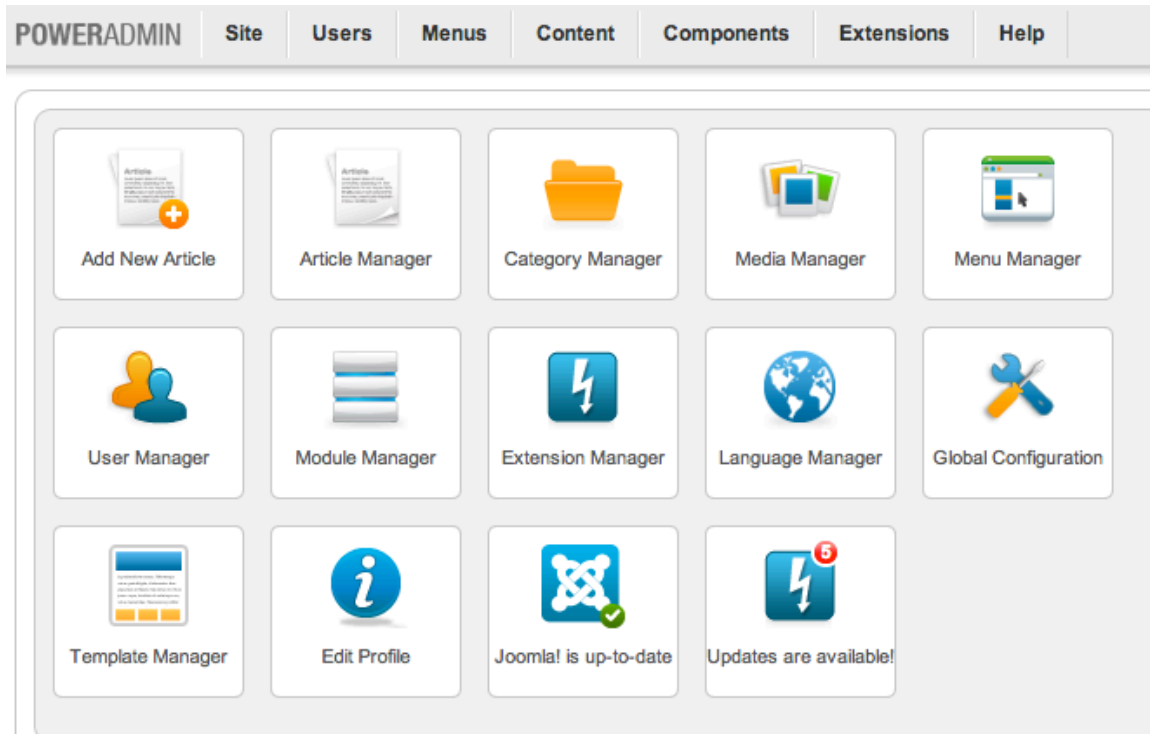


Figure 4.1: Joomla Control Panel

This most important function of this website is to help people to learn some adaptation skills and participate in climate change adaptation. There are several components designed for this website for different purpose. The structure and components introduction of this community adaptation website are illustrated in Figure 4. 2. Then, in Figure 4.3, it is the home page how does the website look like.

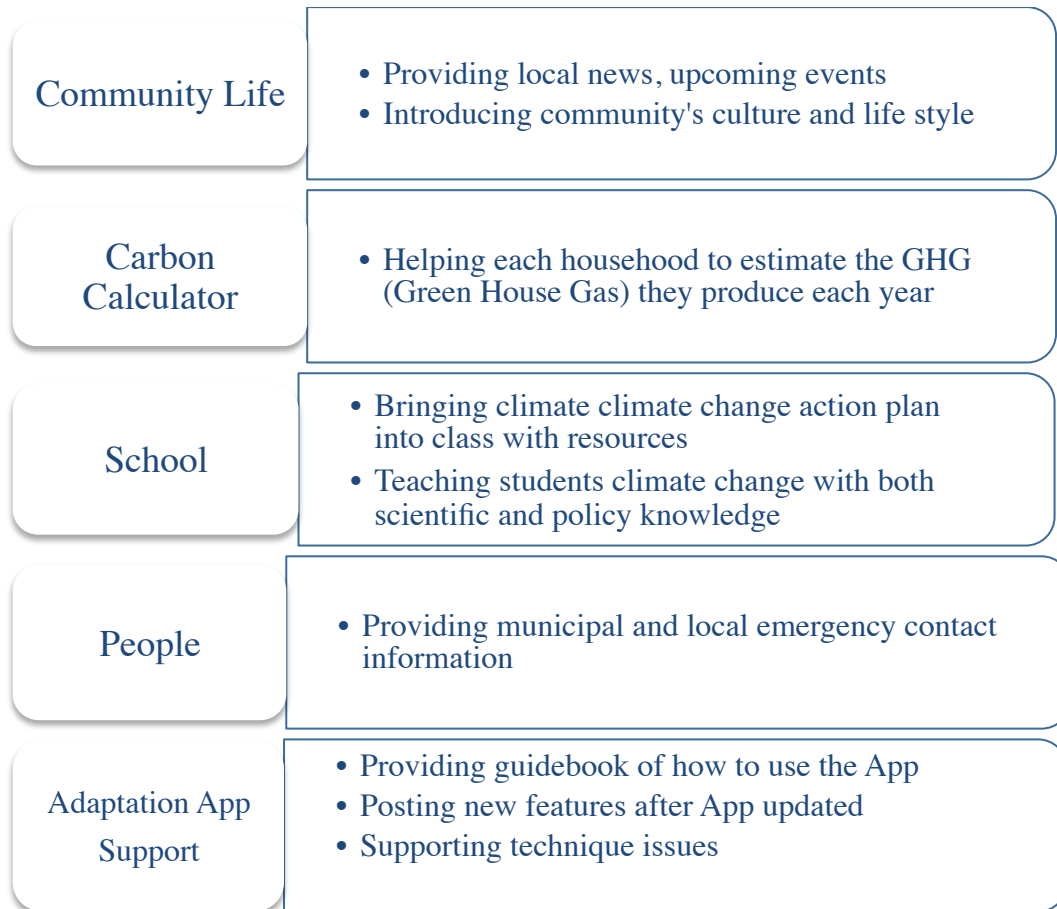


Figure 4.2: Community Adaptation Website Structure

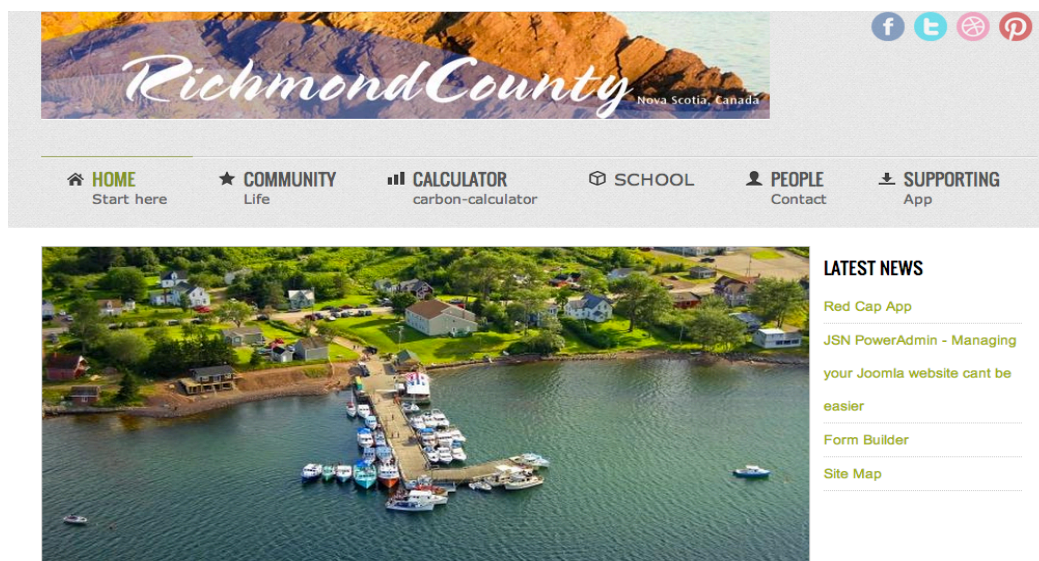


Figure 4.3: Community Adaptation Website

4.2 Community Adaptation Mobile Application Design

Based on the results of analyzing coastal community environmental change emergency response, this chapter develops the processes of designing a real system according to the needs of the adaptation action plan requirements and with the declared position of engaging the community in emergency response.

As discussed in the Chapter 3, in order to build the communication between local government and local community during an emergency event, mobile technologies are useful tools to implement given their well-known presence in communities and extensive use for local communication. The Figures 4.4 and 4.5 below illustrate the basic (1) development environment and (2) fundamental system principles of mobile applications as designed in this case.

(1) Development Environment

As is shown the development environment of Figure 4.1, the mobile device chosen for the development of this adaptation application is the iPad tablet from the Apple Corporation. The reasons for choosing the iPad as the development platform for this application are listed as follows:

1. The 10 inch screen and fast processor inside makes it easy to carry and powerful to compute.
2. With the latest version of iOS 5, it provides users with the world's most advanced mobile operating system and a stable computing environment.
3. The iPad's multi-touch screen allows users to interact with applications more easily and comfortably.
4. With build-in Wi-Fi and 3G, the iPad provides users alternative ways of having access to the Internet.
5. The Application Store is the online application store for Apple devices where users can download thousands of applications. It is also the same place where developers can upload their products by joining the Apple Developer's program. To be able to

develop and upload applications, developers need to be members of the Apple Developer group and to subscribe annually.

6. Xcode is the developing tool used to make iPhone, and iPad applications. It provides file management, code compilation services, and a simulation environment.



Figure 4.4: Features of Apple's iPad Mobile Application System

The advantages listed above, including the large market share of the iPad tablet, make the iPad a logical choice for the application's platform.

(2) System Principles

Before developing a mobile application for community engagement and emergency awareness, it is important to understand who will use this application and how it helps members of the coastal community to take the advantage of iOS mobile application system. As is illustrated below in Figure 4.5, the municipal government of Richmond County, as application controllers, and local community members as consumers of the application information are the two directed users of the specific application that has been developed. When local community users need to communicate with municipal controllers seeking help in the event of emergencies, e.g., due to severe storms, the request for

assistance can be sent through an online cloud server, which stores and manages the mobile application's data flow. As soon as municipal controller receives the request from a local community member, the municipal controller can decide how to provide immediate help and react, at the same time, by sending a notification or advisory information back to local community users to update them about the changing status of the system.

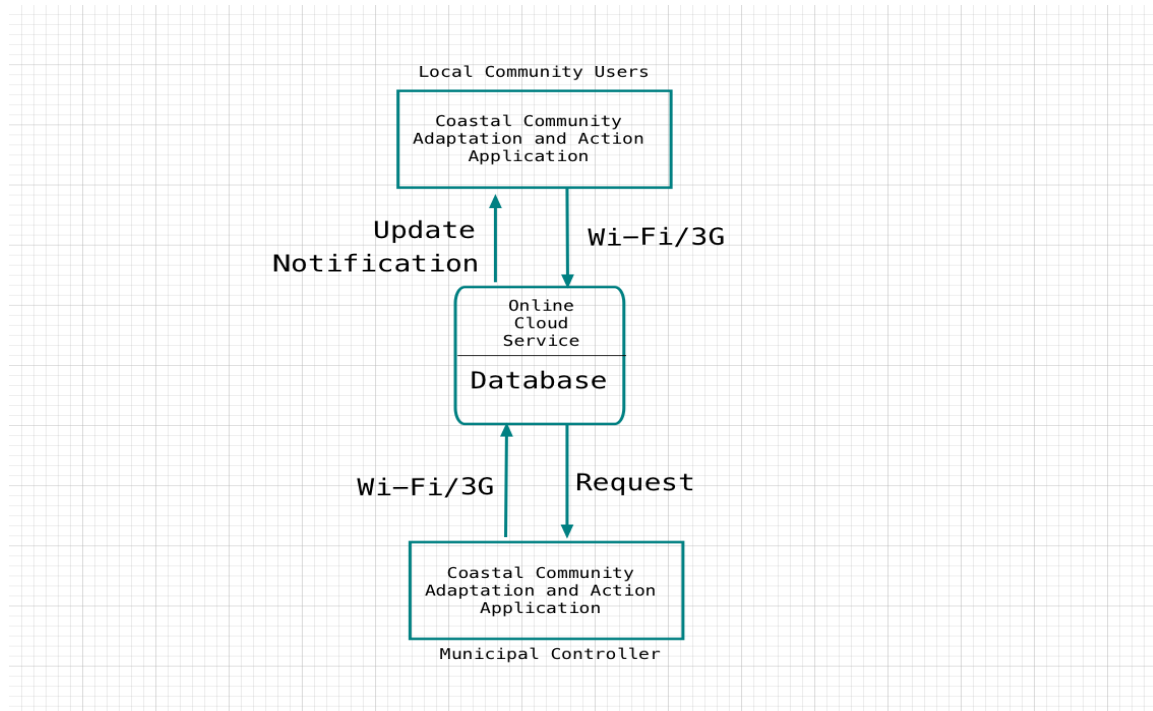


Figure 4.5: Mobile Application System Components for Development and Implementation

The processes of designing the mobile application system can be divided simply into three steps:

- (1) Functional Definition - the designed purpose and use of the application, i.e., for community emergency response engagement;
- (2) Data Management – the data requirements of the system as well as the data use; and,
- (3) User Interface Design – the GUI (Graphical User Interface) design and use for user friendly application including mapping interface, and Municipal Controller and Community User icons and functions.

For the mobile application system, functions must be developed that meet the community users' and municipal controllers' requirements, and ensure proper execution for managing the information and communication links. Thus, the first key step in system design is to define these requirements and develop those functions that meet all users' requirements. Secondly, in order to realize the functions, the relevant data needs to be collected. Thus, the second step in application development is determining how to manage the relationship among the data between community users and the municipal application controllers. Finally, the third step, based on the system functional requirements and well-organized data, is the development of the graphical user interface (GUI) to be designed for the purpose of good interaction between all users and the executing mobile application system. These steps are discussed in more detail in the subsections below toward effective mobile application development.

4.2.1 Functions

The mobile application system needs to specify what such a system could do, and why the coastal community members may need and benefit from it. Usually, it takes more time to define the problem and analyze the customers' requirements, because a system cannot be well developed without a clear understanding of the community's needs and its preferences. In Chapter 3, the process modeling section identified the climate change adaptation and action plan requirements including: (i) the community profile, (ii) threats and vulnerabilities, (iii) the adaptation team, and (iv) elements of action planning. Accordingly, the system functions for the application are defined with respect to each aspect of these requirements noted above. These requirements and their incorporation into the mobile application are presented in further detail below.

4.2.1.1 Community Profile

The community profile has been defined based on the four-dimensional pillars: (a) environmental; (b) economic; (c) social; and (d) cultural (see also Section 3.1.1 above). This profile presents basic information about community assets. The most effective means of presenting community asset information is via a "map" of the community.

However, generally available GIS (Geographic Information Systems), such as Google Maps, do not have enough detailed location information for individual (and small) coastal communities and require further updating for useful detail to be included for both changing coastal climate information, and for emergency response use. Thus, for the community's convenience and needs, it is more appropriate to have a customized mapping system that the community itself can easily and effectively update.

Thus, the first function required by the community is to utilize the community profile information through a Customizable Community Mapping System developed for the mobile application. This mapping system allows the community to edit, modify, and annotate its physical map. The community may want to see different kinds of location information on the map. For example, there may be particular interest in locations related to the environment, e.g., coastal attributes. Thus, a second mapping function includes community profile information that is available as Location Layers with the four different layers based on the four community pillars. The Customizable Community Mapping System (1) and the Location Layers (2) are presented in further detail below.

(1) Customizable Community Mapping System

There are three things that need to be verified before editing the community's own mapping system. As discussed in the Section 3.3 above, this information includes location names, functions and coordinates. Once a place is identified with this information, the community can add it on the map by using different representative icons, which present different location information with names and text information, e.g., to identify banks, commercial outlets, restaurants, public locations such as police stations or post offices. Figure 4.6 illustrates the control panel with icons where the municipal controller can edit the community's map information. The Municipal controller can choose each of those icons and place it on the map using the simple touch screen features of the iPad to update and detail the local community mapping system.

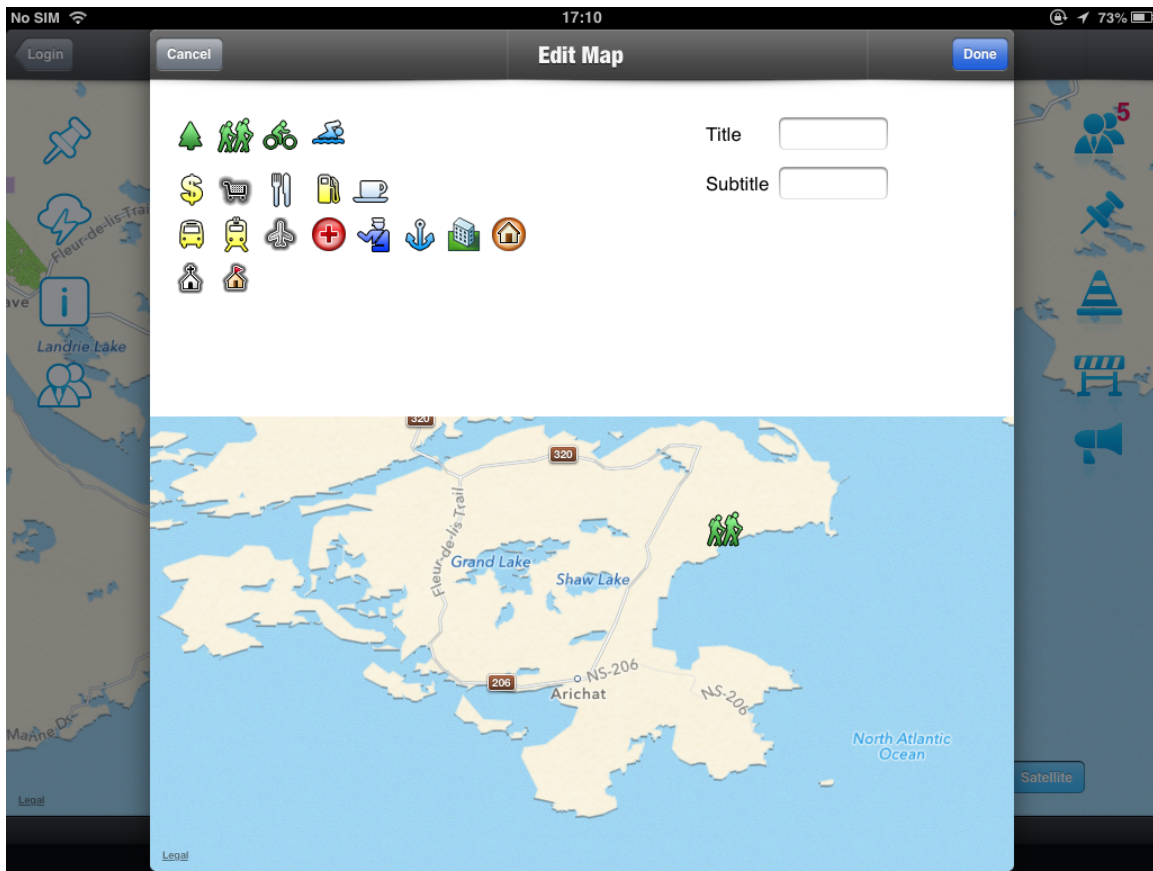


Figure 4.6: Customizable Community Mapping System

By using the simple mapping functions on the iPad, the community is able to develop its own customized mapping system for both adaptation and community development planning use. There are two advantages to this function, which makes it different from other GIS-type applications. The first is that users can add annotations for a location by simply touching the map, and undo it by pressing another position on the map, which makes the editing more user friendly. At the same time, it shows the coordinate value to the users, which makes the editing more accurate. The other advantage of this function is that all the location information has been added or modified on the map are dynamic, which means other users can see the changes as soon as the editing is done. This is what other GIS cannot do, because they do not provide the same mobile, real time interface for multiple users. The iOS gives users this opportunity to download the same terminal from the Apple Store to work on the same map and let other users see the update at the same time.

Location Layers

This function allows users to see four different layers of the location information as defined in the community profile as environmental, economic, social and cultural. By using this function, community planners or research groups can analyze how evidence of climate change, e.g., storms, etc., affects each designated community pillar accordingly. Figure 4.7 illustrates a representative environmental pillar layer with appropriate icons on the map as an example.



Figure 4.7: Community Location Layers

4.2.1.2 Threats and Vulnerabilities

The next process in the mobile application development is the Community Threats and Vulnerabilities. From past adaptation experience and research results, communities have already known which areas are under threat and what kind of threat it is. These areas

have been defined as vulnerable areas in Section 3.1.2 when developing the framework for climate change adaptation. In order to help community members to understand the impacts of climate change and how best to adapt to it, it is useful to illustrate these areas on a map and make some brief description about the impacts. Figure 4.8 shows how the areas information is presented on the community mapping system along with a brief editable text description.

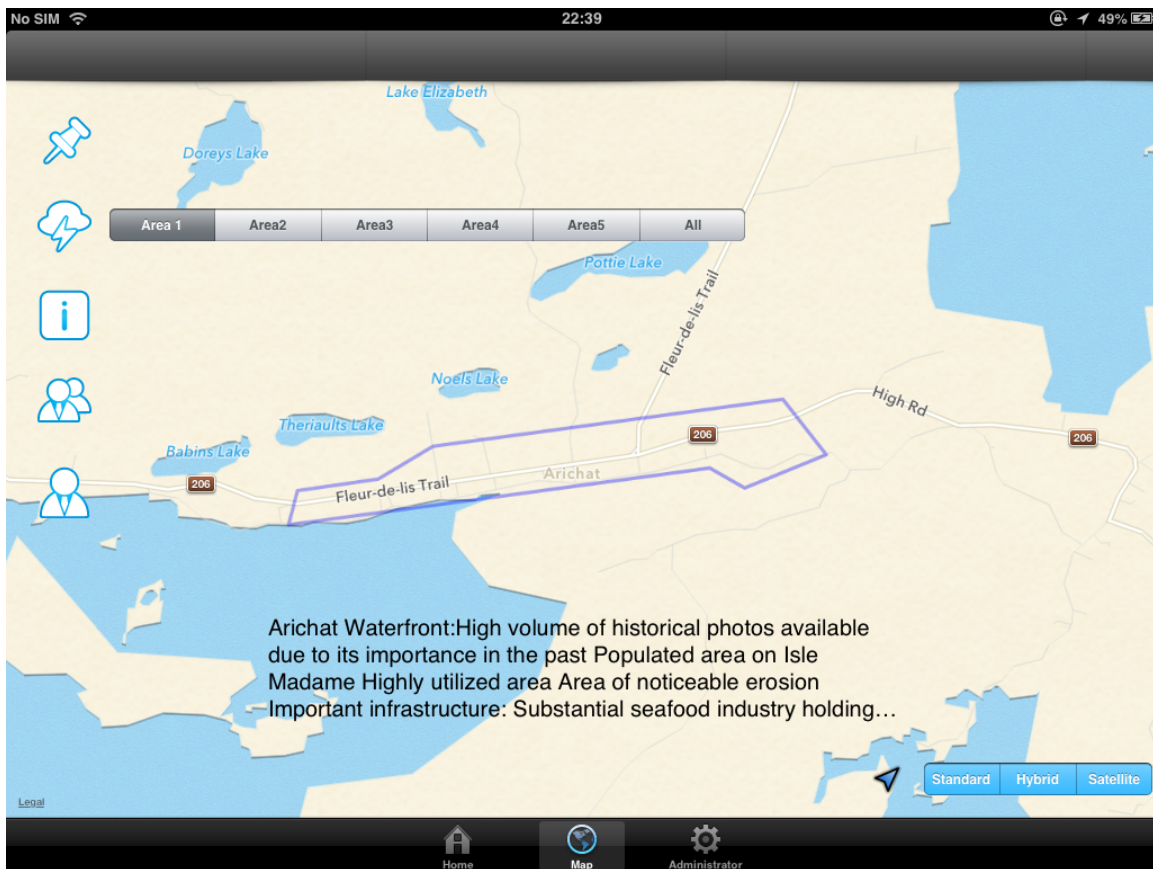


Figure 4.8: Community Vulnerable Areas Indication

This function is named the **Vulnerable Areas Indicator**. Associated with the function in community profile, both municipal and local community users have the ability to see the combination result of these two functions. For example, users can choose one of the community profile pillars to view, then sequentially select the vulnerable area they are interested in seeing to examine which elements in this community pillar are vulnerable in the indicated vulnerable area.

4.2.1.3 Adaptation Team

As an important participant in climate change, the members of the adaptation team (section 3.1.3) play different roles at different levels. The federal government and the provincial government focus more on policy-making and financial support. However, the local government has more responsibility in adaptation planning and emergency response. Thus, the local community is seen in this research and the development of the mobile application as key participants. According to each team's different responsibility and resources, as described in section 3.1.3 above, there are three main activities designed in the mobile application system expressly for the purpose of communication among community members. These include: (1) Policy links; (2) Local government communication and response; and (3) Local community contacts among community members.

(1) Policy Links

At the top level of decision-making, the federal and provincial governments make policies and give support to guide local governments in specific areas of development including adaptation and action planning for changing coastal climates. In general terms, people who live in local communities do not receive or understand enough of this kind of general policy information from higher levels of government. In the mobile app, this function seeks to provide a better way to improve the public policy awareness in climate change by providing easy access to national and international policy information. As such, the mobile application provides access to several national and international policy websites chosen as fundamental links to the changing climate systems. These websites include reference to global ICLEI and UNDP measures, and well as national Canadian policies of FCM, and the MCCAP, together with the federal Emergency Measures program and respective provincial websites.

(2) Local Government Communication and Response

In the context of climate change adaptation and emergency response, local governments have more immediate responsibility to make plans and emergency action response. It is therefore necessary for local governments to cooperate with each department during different situations in emergency. For the convenience of finding the contact information of each local department, e.g., the local fire station, and paramedics, this mobile application function is designed to provide local community users with alternative ways that they can contact the relevant government department or specific responsible people who can provide needed support. Contact information included in this design function contains: the name of the department or person who is in charge, phone numbers (telephone, cellphone), email directories, and physical addresses.

(3) Local Community Contacts

Local community members are important participants in coastal climate change adaptation and emergency response, and it is significant that the mobile application be designed to engage community members in the process. Community members have obvious advantages of immediate response that municipal governments cannot hope to emulate. The Local community's effective response reduces the municipal government's pressure and focus on other areas where more support may be required. Local community contacts could include organizations (e.g., businesses), or faith-based groups (i.e., churches), NGOs or volunteers (e.g., senior citizen groups). These organizations or individuals should be trained before registering. Once they have been proven as helper, their contact information would be stored in a database, which other users have the ability to access. The local contact function of the mobile application would be realized by marking their location and contact information on the map, that other users who need support, can easily find their nearest neighbor and provide immediate assistance as the case need be by using the mobile application contact information.

4.2.1.4 Action Planning

The functions designed above are intended to be available continuously during the mobile application execution. This requires on going updating and modifying relevant information in the system. As such, these activities are made available prior to an emergency event and in “preparatory modes” of operation, i.e., to keep community members informed and prepared for emergency action if needed. In this section, several mobile application ‘action functions’ are designed to assist both the municipal government, and local community members in order to communicate and react during emergency events. In the paragraphs below, action functions for: (1) push notifications, (2) location and status sharing, (3) places advisories, and (4) areas advisories are introduced.

1) Push Notification

As emergency events occur, it is important for municipal government to advise people how to prepare and how to react. Traditional media like TV and radio are widely used as aggregate emergency notifies as in the past. However, with the development of mobile technology, the municipal government has another choice to send individual notifications to the public through the Internet. The advantage of using this technology is that it does not require professional equipment and has no passive or fixed location constraints as with the traditional media. The push notification function allows municipal government to send alerts, advisories and current status information to all the users at any time, and anywhere.

(2) Location and Status Sharing

For people who live in local community, it is important for them to have an opportunity to post their status and, if needed, to send a status message asking for help. By using this function, individual people can post their personal status, e.g., “OK”, or “Need Help”, including a short description of the situation they are in, associated with their current location information. In the designed mobile app, there are four status conditions that

community members may select. In Figure 4.9, it presents the location-sharing window where local community users can share their personal information, location and status.

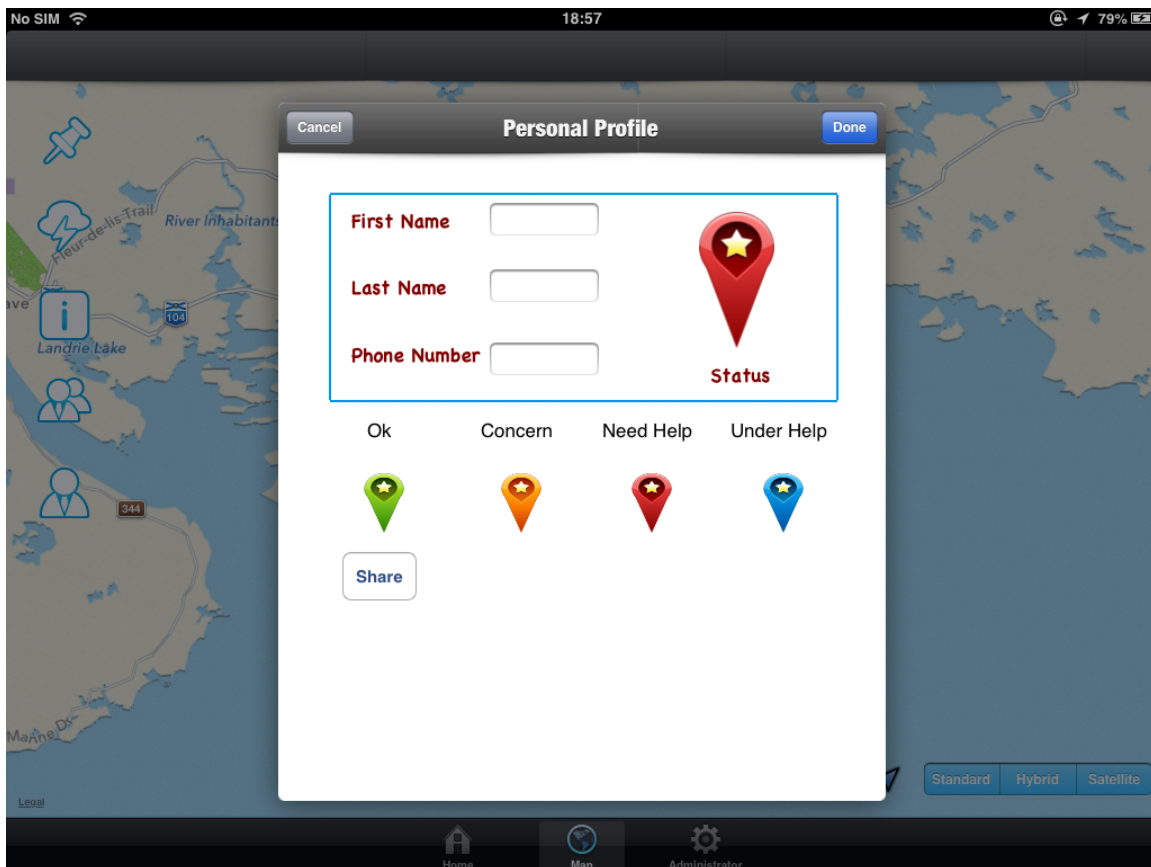


Figure 4.9: Location and Status Sharing Window

They may also choose to let other community members “see” them and their status on the map. At the same time, the Municipal controller would consider each status to make an action or provide support. Once people receive support from either municipal government or helpers, they should change their status or release their locations on the map to update the situation.

3) Places Advisory

Sometimes, the municipal government controller receives the latest information in an emergency situation. At this point, they can use this function to post the places

considered to be dangerous on the map and send this notice to all other users. Figure 4.10 shows the control panel where municipal controller can post advisory places.

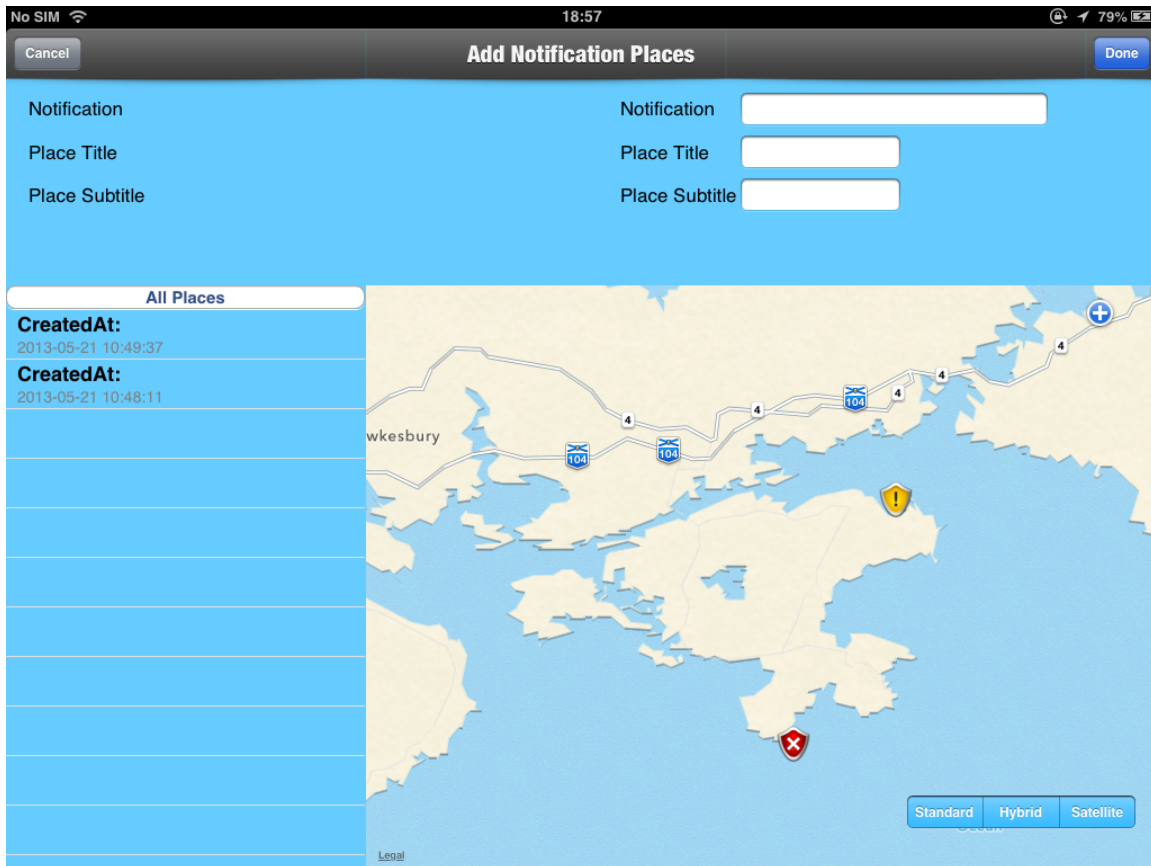


Figure 4.10: Advisory Places Controller Window

The places posted would show on each users' map as a short rolling banner notification as an "alert" to mobile application users. Other times, people in the affected areas would experience the dangerous situation before municipal government receives the information. At this time, local community's status would be used as advisory places by municipal government. This function also builds up the communication between government and community to share and obtain information from each other.

(4) Areas Advisory

Similar to the places advisory function discussed above, areas advisories are designed for the municipal government controller to post a dangerous area on the map with

notification, and then send this notice to all the mobile application users. Figure 4.11 presents the interface, which allows municipal controller to post advisory areas information by drawing an area on the map. They can also manage all the areas posted by using the table list on the left side.

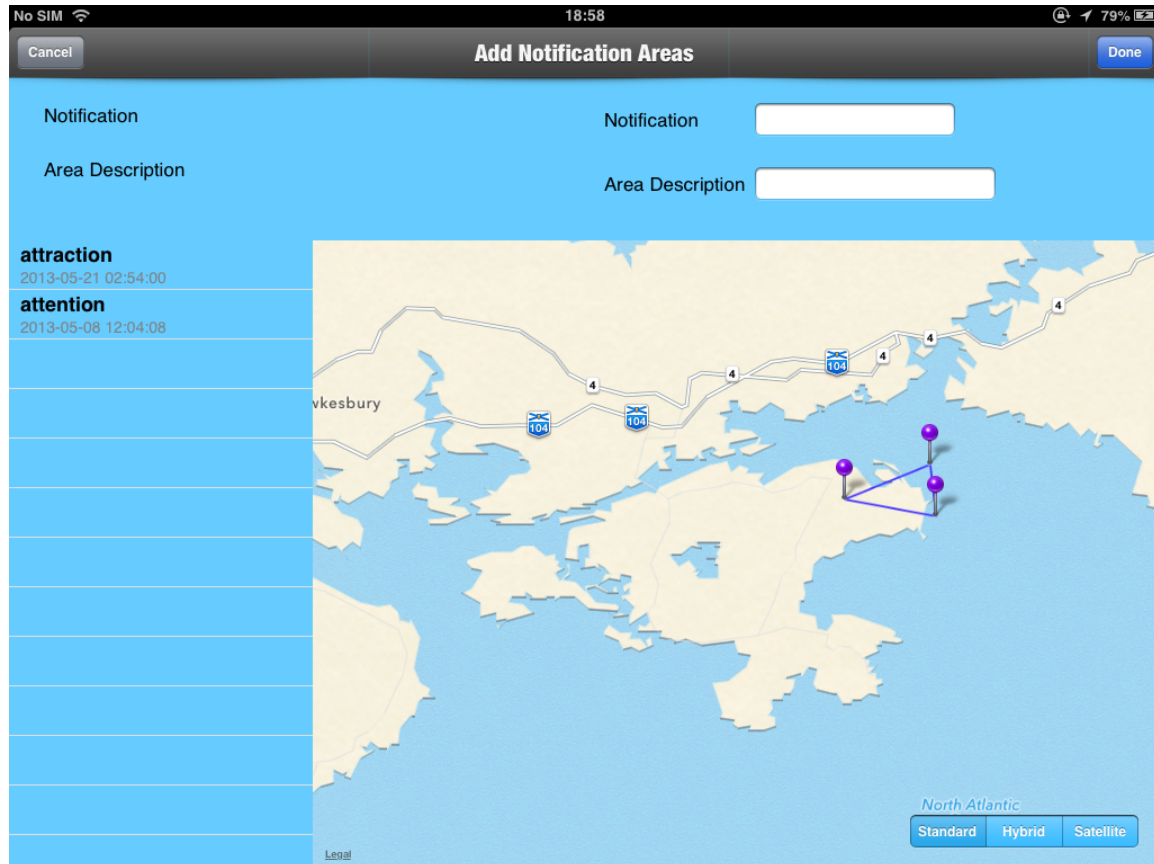


Figure 4.11: Advisory Areas Controller Window

By receiving this area information, community users would understand clearly where they are not permitted to go. The difference between “places” and “areas” is about the density. Some advisory places are only single points on the community map, which do not affect other places around. In some situations, the advisory places keeps increasing in a certain location, which becomes an area. It would indicate that the municipal government and this area probably require more help and support.

In summary, the above main action plan functions in this climate change adaptation system are related to geographic information, which presents the advantage of mobile

technology. The main purpose of the design of a location related system is that it allows both municipal government and local community to engage all community members easily to examine and understand the impacts of the changing environment on people directly. The following figure, Figure 4.12 summarizes all the functions in the mobile application system.

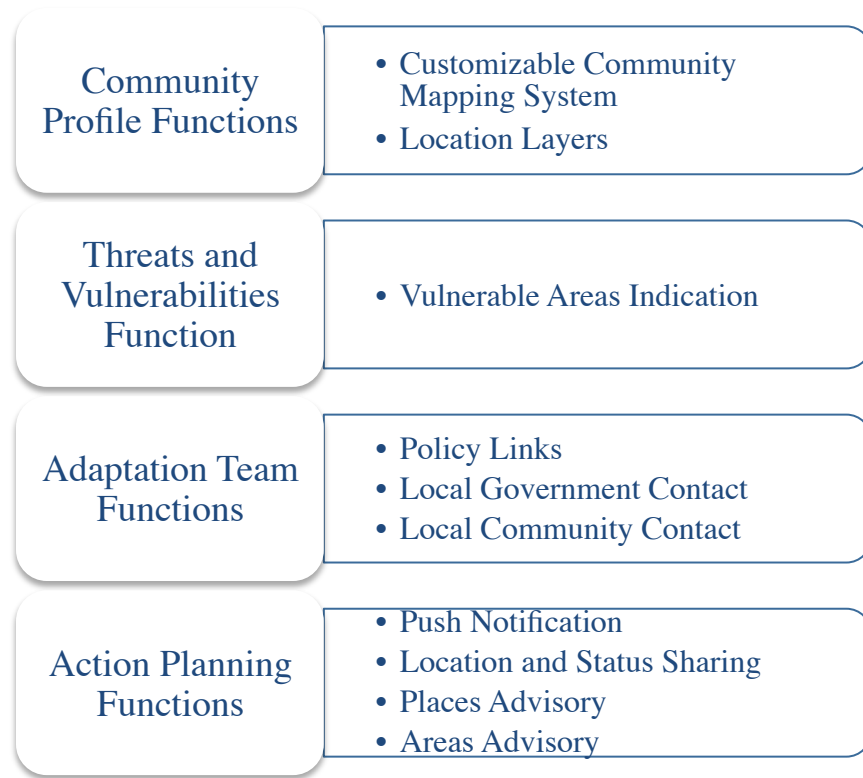


Figure 4.12: Mobile Application Functions

4.2.2 Data Management

The next important step in system design is data management. Once the main functions have been defined, they require relevant data as input to make each of the function to execute as intended. In the case of emergency response, the concept of designing this system is based on location information. It would be useful for both municipal controllers, and the community users to understand what is happening, and also to know where it happens in the community. The data in this system are real time. These data are stored in a server called *Parse* (www.Parse.com). *Parse* is a mobile application server

provider. Thousands of applications are using *Parse* as data storage and management because it has a very powerful APIs (Application Programming Interface), which allows different mobile platforms to run. For this research, *Parse*'s APIs are used as the interface to upload, download and query data from the municipal controllers and the community users. According to each function's requirement, three steps are required for data management: (1) Data Collection, (2) Data Input, and (3) Data Updating. These elements of the mobile application are introduced in the following paragraphs.

(1) Data Collection

Before collecting data, it is important to know what kinds of data are required in the system and where to find the information. There are two kinds of data collection in this system, which are data collected by community members in real time, and data collected by the system itself for distribution to application members. This information is stored in the database for further scientific or research analysis use. Based on each function's requirement, the data that needs to be collected is listed as follows:

- a) *Featured Locations*: For the purpose of climate change adaptation, communities should pay more attention to some locations, e.g., coastal roads, which are considered to be important as introduced in community profile.
- b) *Vulnerable Areas*: It is important for both the municipal government and local community members to understand those known areas where environmental change is already in affect. Communities acquire this information based on historical data, adaptation experience, or research results.
- c) *Policy Information*: As mentioned above, providing an easy way for the people who live in the community to get the latest climate change adaptation policy information would improve the public awareness and participation. This information is obtained from relevant national and international websites, as developed in Section 4.1 above.

- d) *Contact Information*: The contact function in the mobile application system allows users to find easily emergency contact information. There are two kinds of contact information in this system, namely, (i) municipal contacts and (ii) local contacts. Municipal contact information is collected from each local department who has the responsibility to respond during an emergency event, EMS services. Local contact information is from the organizations or individuals who act as volunteers when an emergency event happens, e.g., church groups, senior citizens groups. These organizations register their profile information and then are updated in the system over time.
- e) *Advisory Information*: The most important function in this system is to allow municipal government post notification and advisory information. It also gathers the latest users' status information if someone shares their condition. To provide advisory information, the municipal government mobile application controllers can either use reports sent from emergency response department or verify the condition information shared by users.

(2) Data Input

Once the required information of the forms noted above are collected, the municipal government mobile application controller can input these data into the mobile application system. There are two kinds of input data, namely, static and dynamic data. Static data means the data does not change often and includes information such as known vulnerable areas, and municipal emergency contact information. These data are input once-and-for-all (updating included) and do not have to be regularly modified. For dynamic data, the municipal government mobile application controller uses the current dynamic information as resources to monitor, e.g., community members' status. This dynamic data also includes community profile information, and local contact information.

(3) Data Updating

In order to provide real time information to community users, the municipal government mobile application controller has the responsibility to post and update the current conditions and emergency situations. The 'Push notification' function is used as an emergency alert to each user for indicating the upcoming event. The municipal government should also keep tracking the latest event information and be ready to send notifications to community users. When the municipal government receives request from the user who shared their location and show critical conditions, the advisory places and areas information should be updated according to how many users are having the same issue.

4.2.3 User Interface Design

When mobile application users launch the application, a well-designed user interface determines easy and convenient use. Touch screens dominant most smart phones and the tablets' market, e.g., iPads, and it has totally changed the way in which mobile application systems are designed. Traditionally, people have to interact with computers and cell-phones through a keyboard or a mouse as input tools. These devices actually slow down the reaction period and it takes additional computing space to manage these devices. With the touch screen, users do not need devices to input data, or to navigate to a target. Instead, all that is required is a touch, thereby changing the way users interact with applications, and giving users a different experience. Touch screen capabilities also challenge the design of mobile applications since the interface has to be clear, well organized and easy to use. In order to achieve this goal, the system designed in this research uses a setoff touchable icon to represent each function, and to define the different views of the application corresponding to its different purposes. In the following section, the concept of designing the touch screen mobile application system is introduced in three parts: (1) elements, (2) layouts and (3) views.

4.2.3.1 Elements

The elements in this mobile application system present each of the active parts on the screen with which users can interact. All the functions defined in Section 3.5.1 above need to be presented as a screen element in the system in order to define the touch screen interface so that the system can respond to users' requirements. There are several objects in iOS that are used for the interaction between the user and the application. These are denoted as "buttons", "tables" and "gestures". Each of these objects is implemented for different functions in this system and the detailed design, and is described in further detail below.

Button Design

For user's common perception, images can describe more things than text and leave a more sustained memory in the user's mind. For the purpose of user-friendly interface design, the buttons used in this application are customized different icons to represent different functions. The following paragraphs introduce each button's meaning and illustrate the actions after clicking on each of the button.



: This icon represents the collection annotations that indicate important location in community wide. By clicking this icon, it shows four pillars options as defined in community profile as four different annotation layer and one option which show all the annotations together.

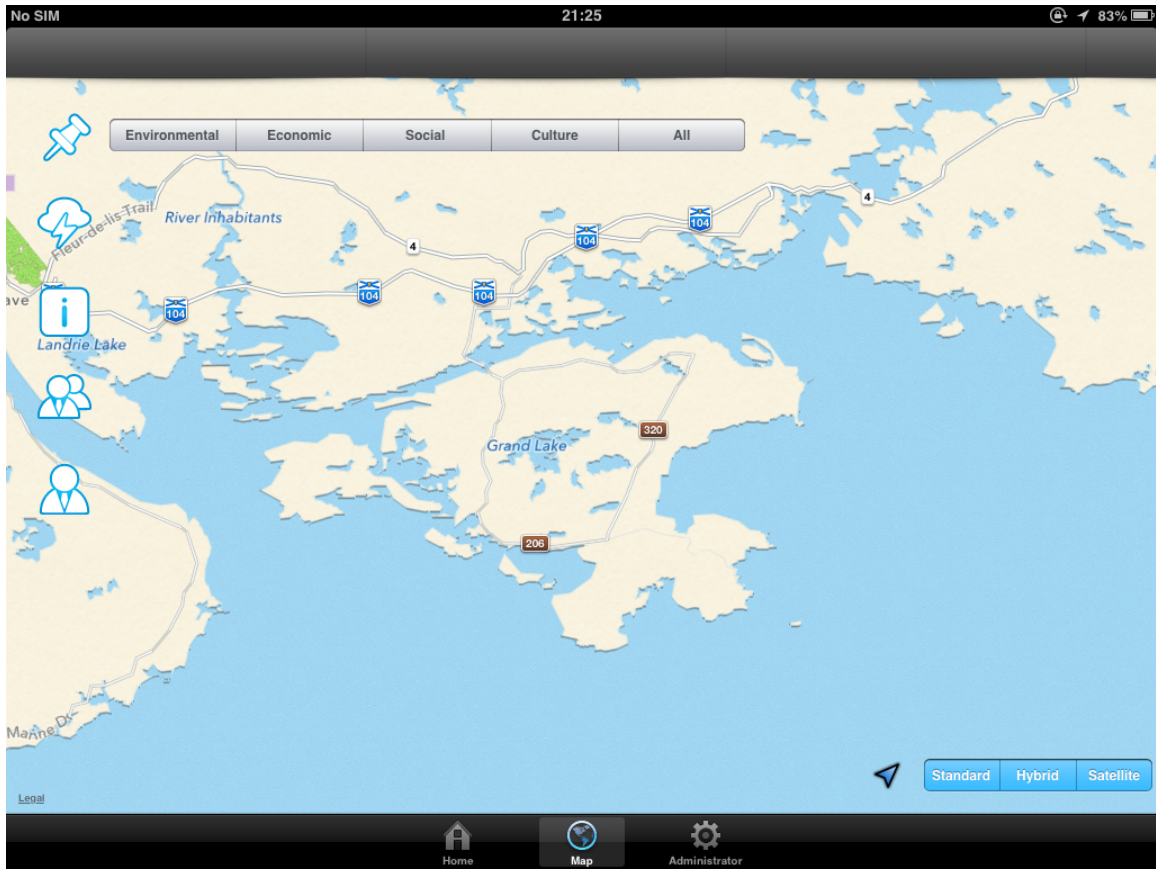


Figure 4.13: Annotation Button Action



: This icon implements the function of displaying vulnerable areas and showing a short description. This icon gives five options for each area, which are pre-defined as static information in the system.

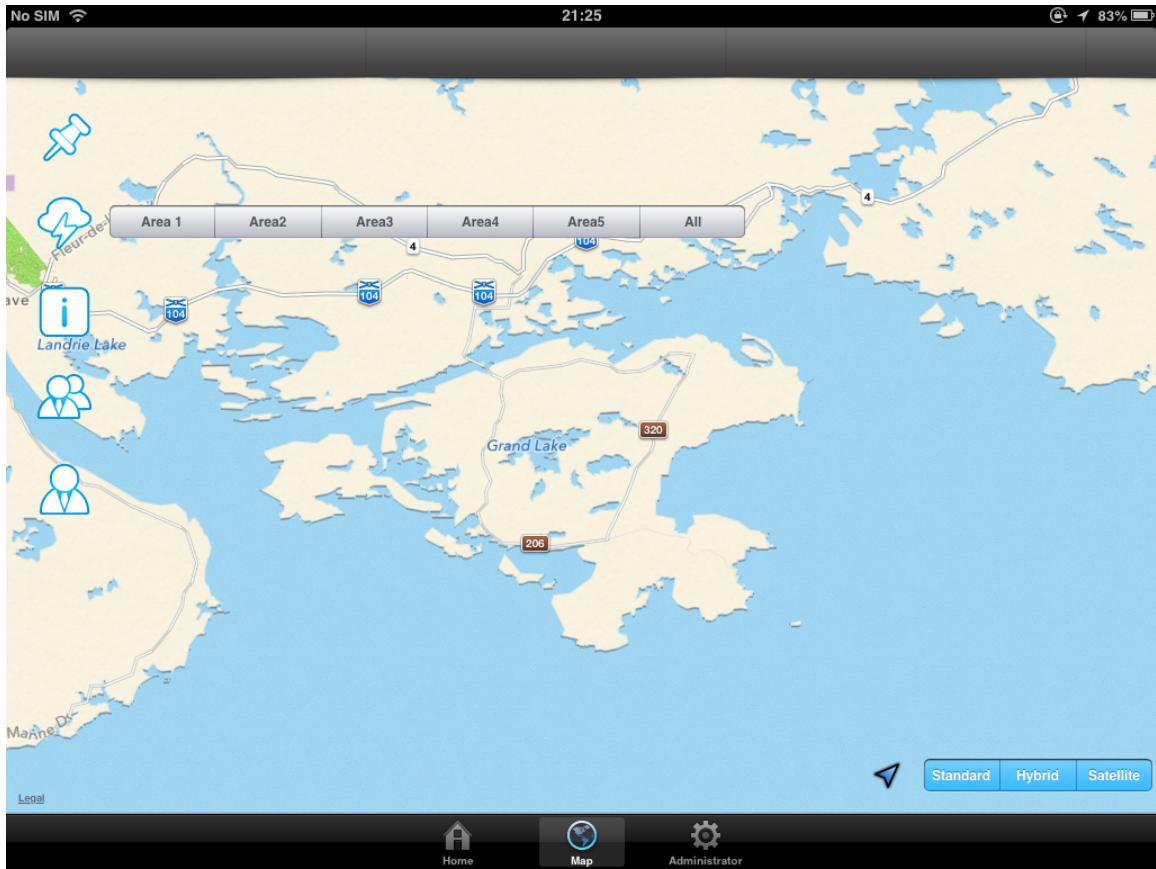


Figure 4.14: Vulnerable Areas Button Action

 : For the function of advisory places and advisory areas, this exclamation sign indicates the users that this is something they have to pay attention. By choosing each option, the places or areas would be marked on the map.

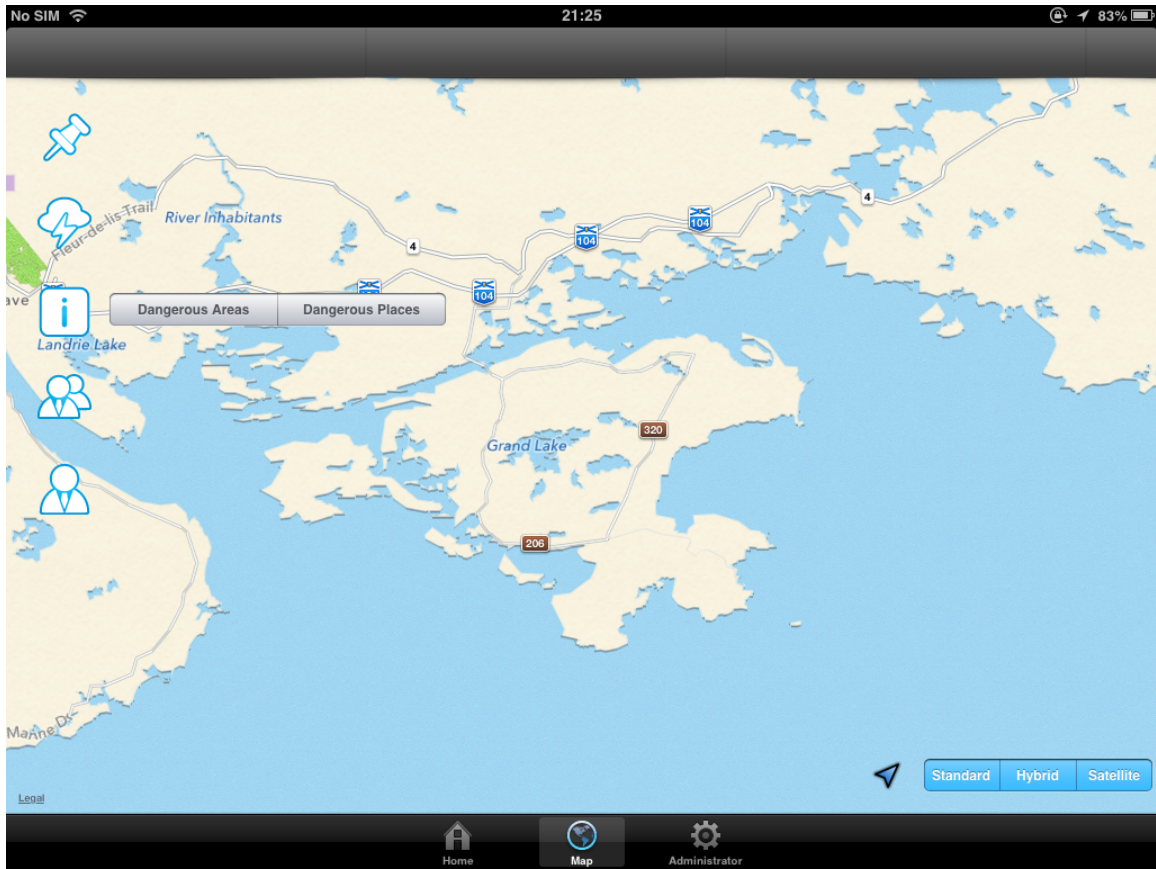


Figure 4.15: Advisory Information Button Action

 : This icon is commonly used in other applications indicating that sharing information or showing the users in the same group. For the function of finding out who need help and who can help during an emergency event, this icon would be clearly understood by users.

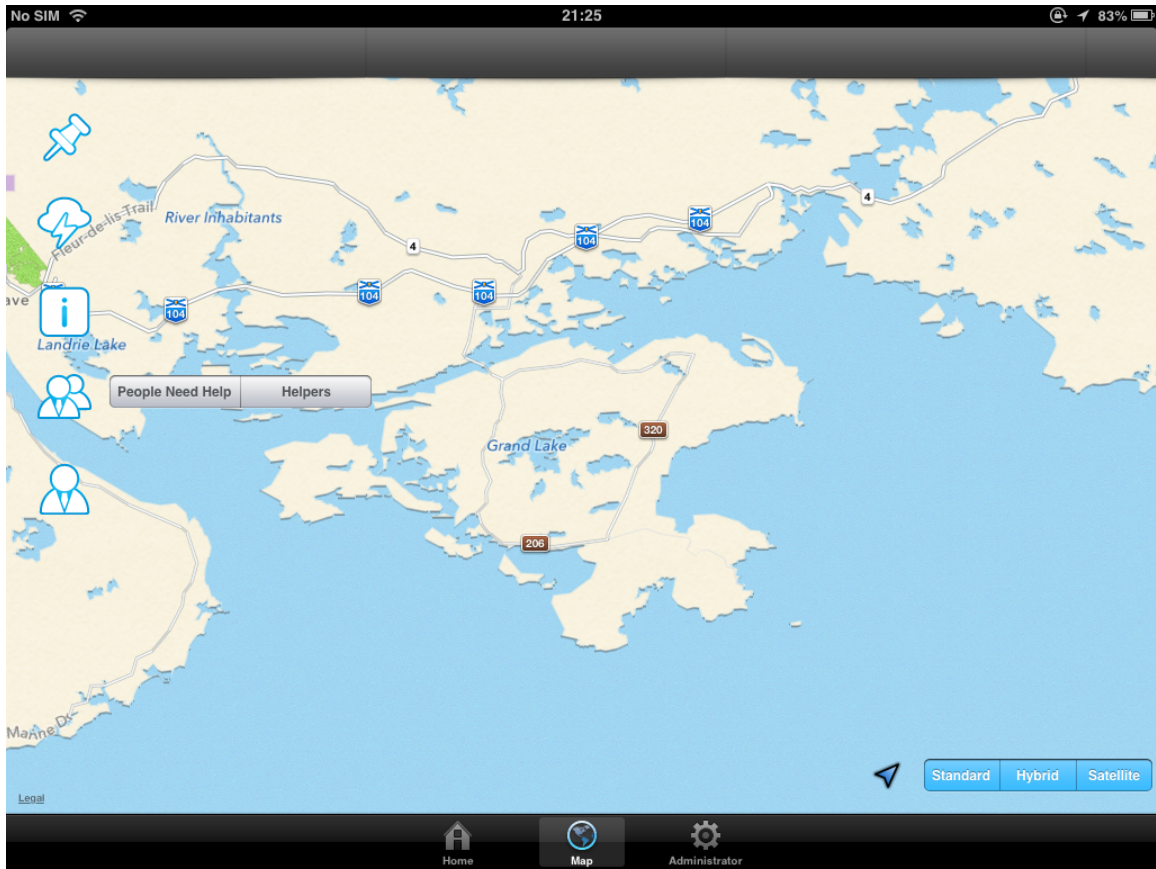


Figure 4.16: User Locations Button Action

 : Similar like the sharing icon, this single person icon means some personal information. In this application, it lead the users to input some information like name, address and contact number in the situation of emergency that users can use this function the share their location and status to notice municipal government and local helpers.

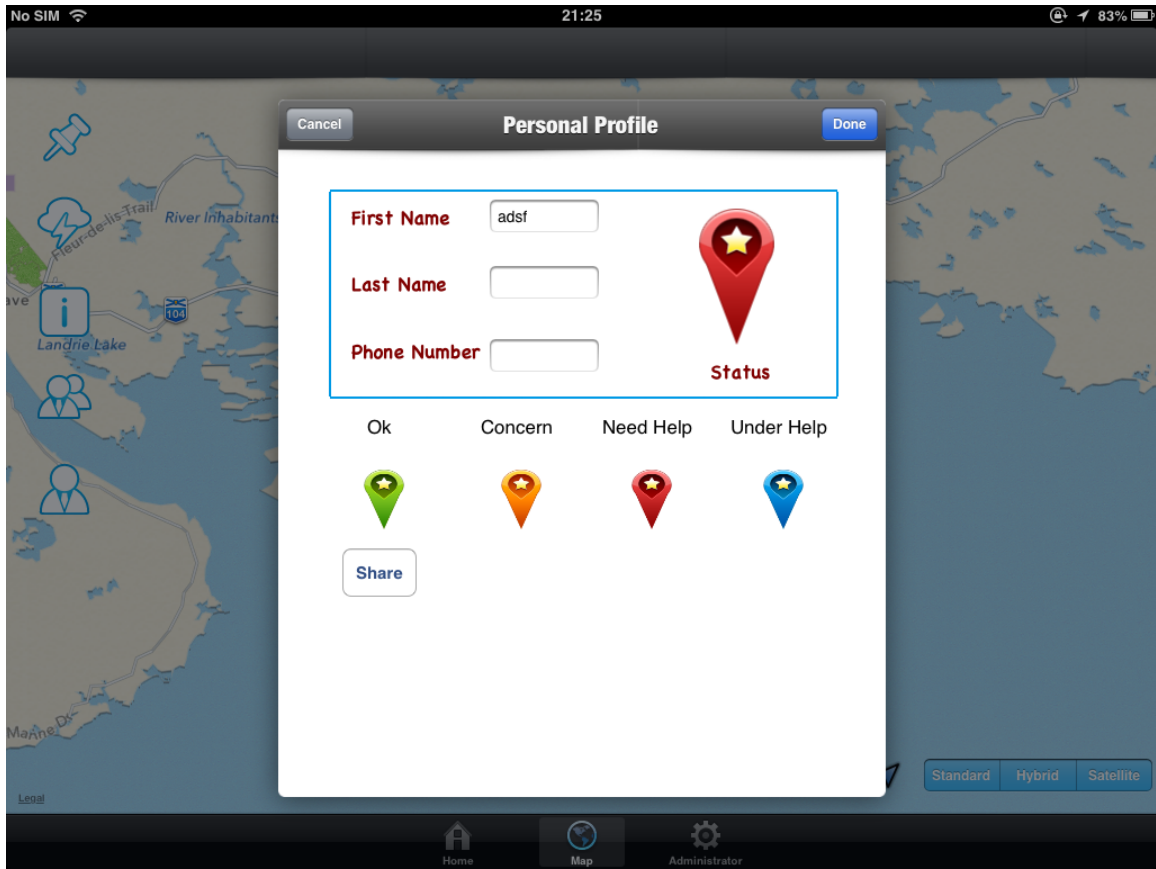


Figure 4.17: Location Sharing Button Action



: This is a similar icon like the first one, which shows the different annotation layers, but different color. The first one only allows users to see each layers of annotations, however, by clicking this icon, users can add new annotations on the map to let other users see them.

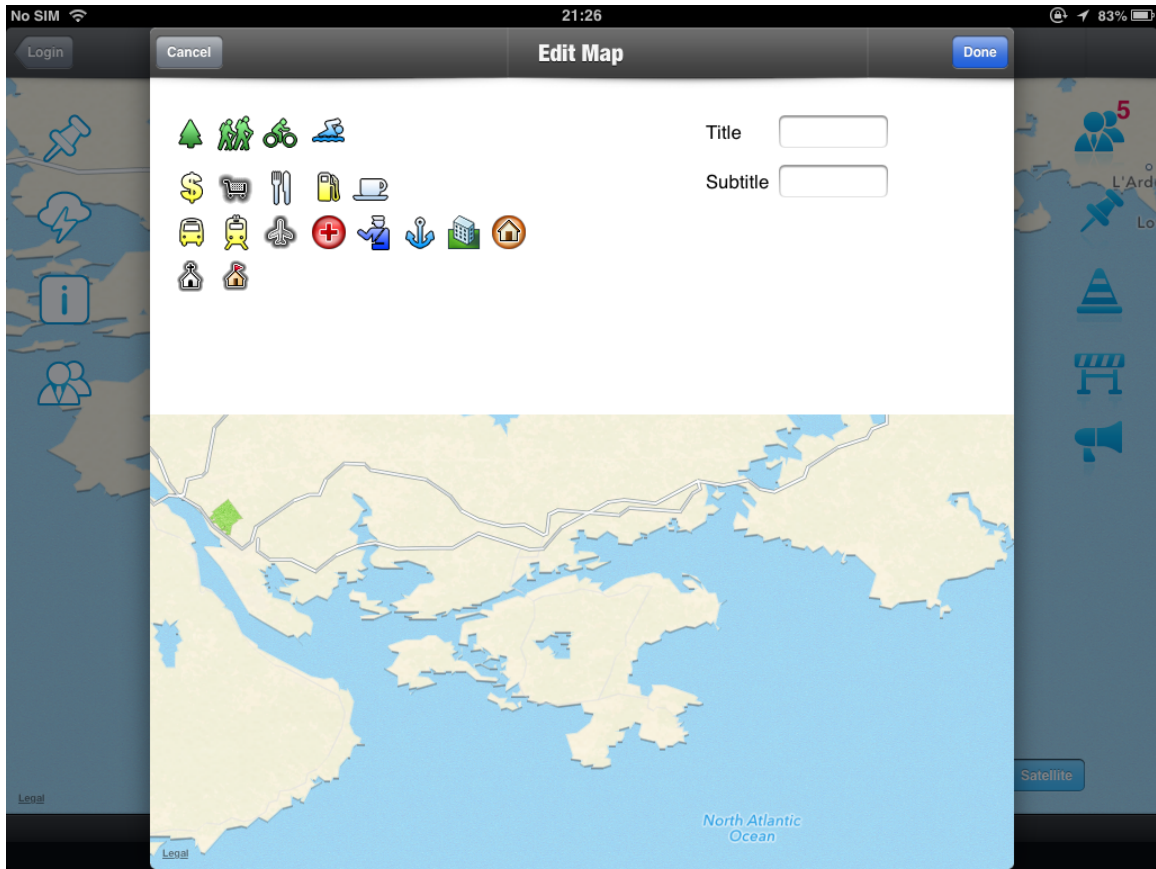


Figure 4.18: Editing Map Button Action



: This icon indicates the total number of users who have shared their locations.



: This barrier icon leads the administrators to a new view that allows them post an advisory place on the map where considered to be dangerous and send notifications to the public users.

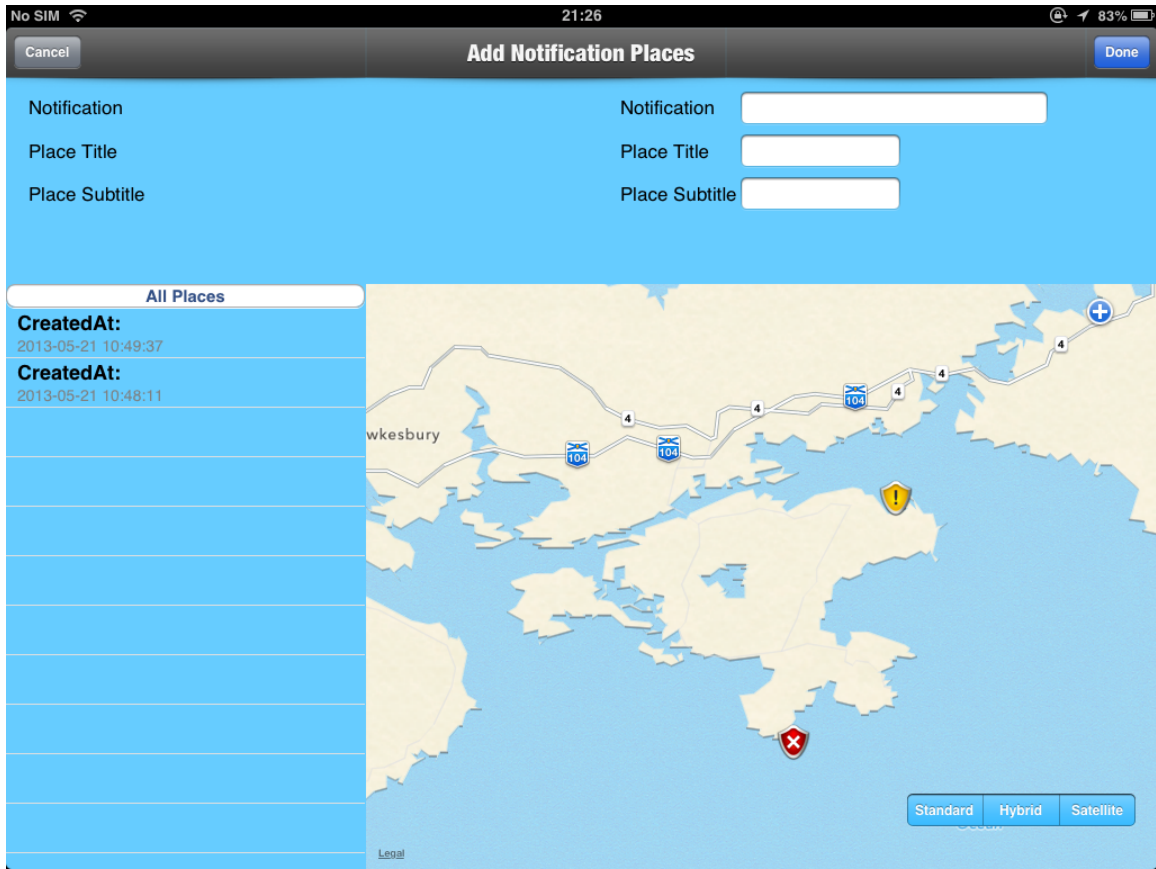


Figure 4.19: Editing Advisory Places Button Action



: This is a similar meaning like the barrier one, which administrator users can use to draw an advisory area and post it to the public users. Municipal government can use this function to mark an area as dangerous and let the local community aware of it.

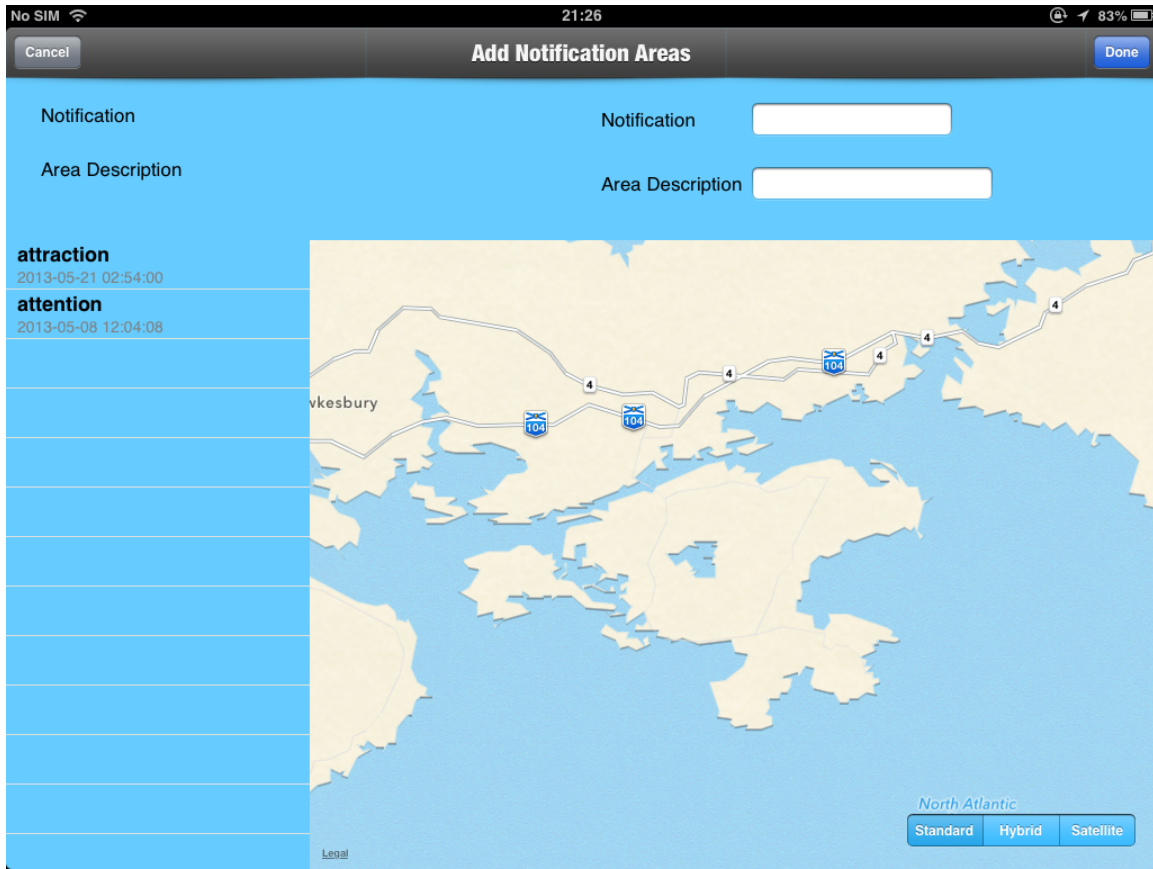


Figure 4.20: Editing Advisory Areas Button Action

4.2.3.2 Views

The screen “views” in the application are used for displaying results or performing an action. Followed this concept, three main views are designed in this application for different purposes. These three views are: (1) Home View, (2) Map View, (3) Personal Status Input View, and (4) Administrator View. These views are also associated with connections to other views and sub-views. The detailed design and layout of each view is introduced in the following paragraphs.

(1) Home View

The Home View is the first screen that appears to users when they launch the mobile application. The application is entitled Richmond Emergency Defense Climate Adaptation Plan or known as REDCAP for short. The application for REDCAP features a

map that is protected by the ‘Red Cap’ as a symbol of the application title. As such, this screen is important in grabbing users’ attention. On the top of the Home View screen, a notification ticker with a bright color shows the current notification news about advisory places and areas, alerts, and other information that the municipal controller needs to replay to community users. By clicking this notification ticker, the application links to the next view, which is the Map View screen, where users can check where the places and areas on the map are that have given rise to the notification. Also on the Home View, there are also three buttons on the right side directly link to News, Policy and Event website, which we have already designed in section 4.1 as a components of community adaptation website structure. The two banners on the bottom correspond to the participating municipal government website and the C-Change project website.



Figure 4.21: Home View

(2) Map View

This is the most important view in this application and where users are expected to spend most of their time checking location information, advisory information, and sharing personal and community member status under emergency event situations. The icon buttons with indicated certain functions are listed on the left side. The first four icons functions show the different layers of the annotations on the community profile map, displaying vulnerable areas with short description, marking advisory places and areas, and showing users who have share location and status on the map. Each of these icon buttons comes with an animated hidden option list, which would show up when users touch the icon as is shown in Figure 4.22.

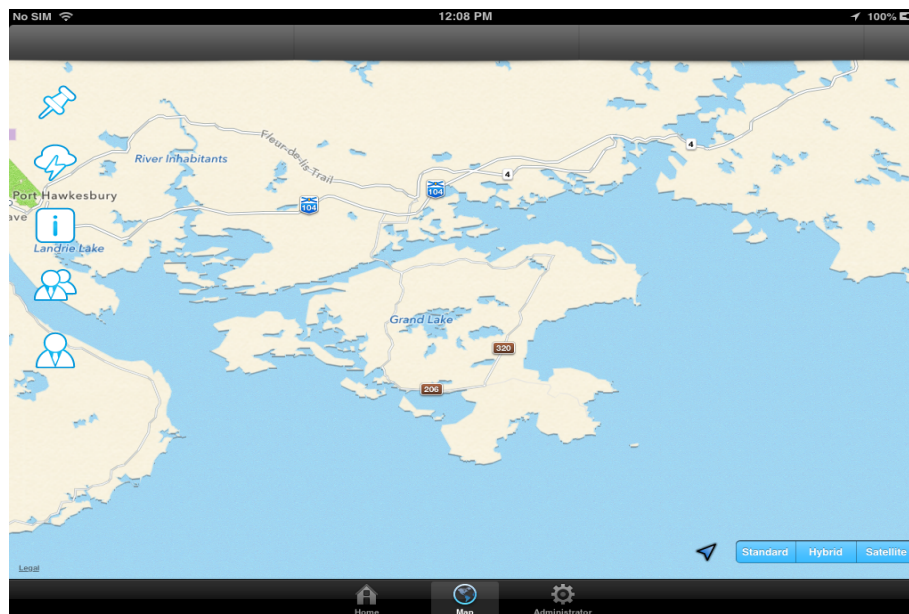


Figure 4.22: Map View

(3) Personal Status View

The last icon is for users to share location and status. By touching this icon, a new sub view appears, which requires users' basic personal information and current status option.

Public users can use this to share and change their status and let other users see them during an emergency event. This sub view is as shown above in Figure 4.17.

(4) Login View

Figure 2.23 shows the view that allows municipal controllers to login as an administrator. Because of the emergency use of this application, only the people who has been authorized can have the right to login to do the editing and posting functions.

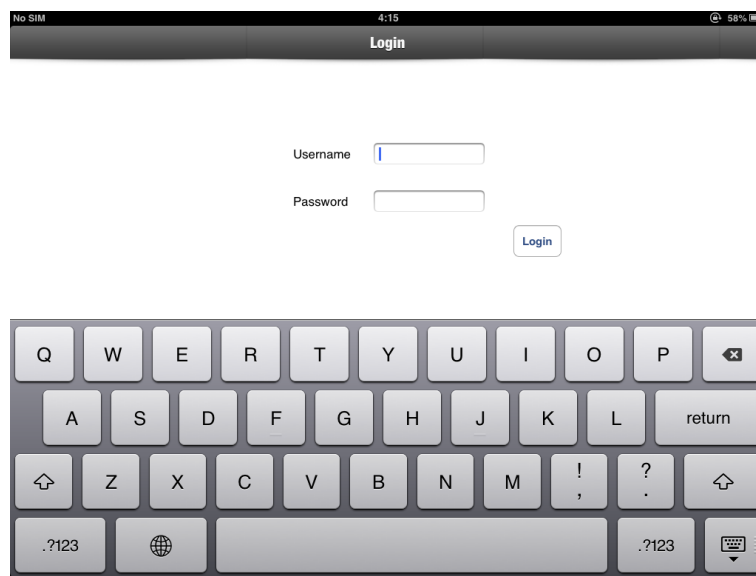


Figure 4.23: Login View

(4) Administrator View

The Administrator View contains the full functionality of the Map Views, but is reserved for the Municipal government mobile application controller group who has responsibility for providing and interpreting application data for release. The Administrator View has advanced screens for the purpose of editing the map's annotation, posting advisory places and areas, and preparing and sending notifications to community mobile application users. This view is only for the authorized users in the municipal government who is in charge of emergency response or reaction. Each of the button icons on the left side of this view comes with a sub view where the administrator user can edit, modify and update

information. The first sub view is map-editing view, which allows administrator users to add annotations on the map according to the four profile pillars. Once the edit is done, the application would update the database in the server and all other users would see the new update. The second view as shown is designed for post advisory places with short notification that tells all the other users that places or areas are considered as dangerous. The third icon corresponds to the third view, which has the function of posting advisory areas for the public users. Administrator users can simply long press the map to draw a polygon area on it and send it with notification to all the other users.

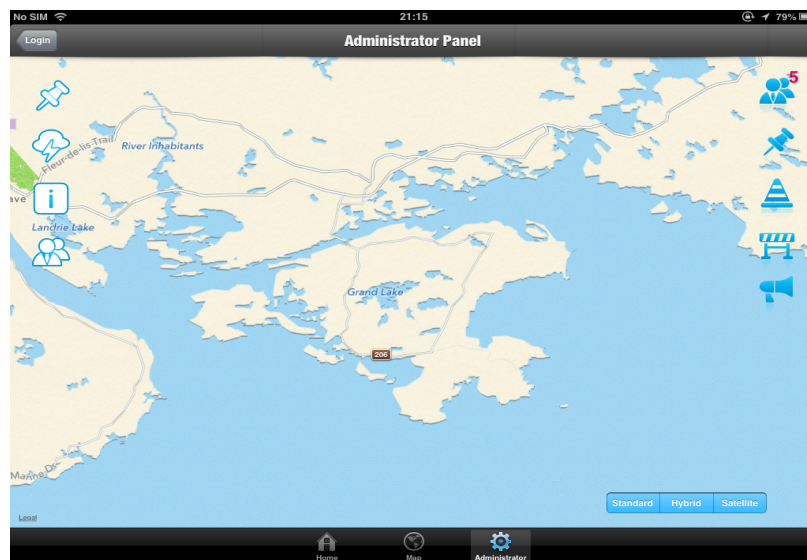


Figure 4.24: Administrator View

4.3 Validity and Verification

Once the steps of defining systems functions, and users' interface design are complete, the process of developing the application begins. In order to realize each function and make each component work properly, an appropriate programming model and testing plan are required. In this validity section, according to the system's requirements and each desired functions, programming methods need to be defined to realize the functionality of the system. Some methods or solutions may not be efficient, because they have to be sufficient and work properly with other components in the system. A well-planned verification process is used to test and find the potential problems, which would

affect the system use and effectiveness in the long run. In the following two sections, the model programming language and the robustness of the application is discussed.

4.3.1 Programming

For iOS application development, Xcode is used as the developing tool to program, manage files, and construct models (see also section 3.2 – System Analysis). The programming language used to develop the iOS application is Objective-C, which is an object-oriented language and powered by many frameworks. The principle of developing an iOS application is based on a Model, View and Controller (MVC) structure. The ‘Model’ is the application itself with functions, methods and classes. ‘View’ tells us what the model would look like in terms of an application. ‘Controller’ presents how the application responds to users’ actions.

In the case of this application development, each of the views introduced above has their own view controller, which has been designed for specific functions. Variables and methods are also defined in the view controllers. There are eight total view controllers designed in this application’s model with different file names and programming methods. These view controllers are listed in the following paragraphs.

1. Home View Controller

The purpose of developing this controller is to show notification text with animation and updated information from web server. There are two places where notification is generated, which is by updating advisory places and advisory areas. Each notification is sent by administrator users and stored in a web server with other relevant information. The notification area on the top of the Home View (Figure 4.21) keeps checking the most updated notification information with the modification time, which public users would clearly understand. The rest of the components on the Home View provide different links to the website for new, policy and event information.

2. Map View Controller

This controller allows all the users the check annotation information stored in the web server. For both research and public use, one of the important functions of this application is that it allows administrator users to update and modify local community's map system by adding and grouping the locations defined in the community profile. In the Map View (Figure 4.22) users can display annotations by clicking the annotation icon and choose the pillar's they want to see. Each annotation is stored in the server with the title and subtitle information. Title describes the name of the location and subtitle gives a short introduction.

3. Login View Controller

In order to identify who are the administrator users, Login View (Figure 4.23) requires a user name and password for navigating to the administrator's view. Only the one who has been verified by municipal government can be registered as administrator user, because some of the functions are emergency related that not everyone should have the right to access. Once someone has been proven as administrator user, a user name and password is created in the web server's database.

4. Administrator View Controller

This is the controller that municipal controller uses to update map and notification information. This view controller has the same functions and methods the map view controller has. There are several additional functions only administrator users have the right, which are indicating the total numbers of people who shared their locations, edit local community's annotation information, send advisory places and areas notification. All of these additional functions come with a new view with their own view controller, which are the Map Edit View Controller (Figure 4.18), Add Places View Controller (Figure 4.19) and Add Areas View Controller (Figure 4.20). The details are discussed respectively in the following.

5. Map Edit View Controller

This is a sub view of administrator view controller shown in Figure 4.18. It allows administrator users to customize and update annotations on the local community's map. In this view, user can long press on the map and the application convert the coordinate on the screen into the map coordinate, then add a simple annotation on the map. After that, users can choose different annotation icon on the top to customize it. At the end, they have to input the title and subtitle of the annotation and press done to update the map. All the information including the coordinate, icon of the annotation, title and subtitle would be stored in the web server by press the done button. Once the users want to see the update, they can go to the annotation section mentioned above to see each pillar's update.

6. Add Areas View Controller

As shown in Figure 4.20, this view provides the administrator users the ability to post an advisory area on the map that public users can see it and pay attention in the situation of emergency. It is also a sub view of admin view controller. There are three parts in this view that allows the administrator users to view, edit and update the area information. First, the map in this view is the part that administrator users can draw a polygon area. Users can just simple long press on the map to add each point that can form an area. A method in the program would add a line automatic after a point is created. The second part is the two-text field for inputting the area information. One text field is for the notification information that needs to be displayed on the top of the Home View. The other one is for a short description of the area situation. Both of these text information along with the area coordinate would be sent to the web server once press the done button on the top right. All these information would be stored in the database, which public users can access. The last part in this view is a table view that list all the areas posted. Users can choose each cell in the table to see the detailed information about the areas have posted. One of the table view's attribute allows users to swipe and delete the content in the table, which administrator users can use this function to delete the areas no longer need to be considered as dangerous.

7. Add Places View Controller

The last sub view in the Administrator View controller is adding places view shown above in Figure 4.19. It is similar like the add areas view controller, which generates advisory places instead of areas for the purpose of notification. It also has three parts in this view. The main part is a map view that administrator users can add advisory places by press the plus button, and then three different icons, which represent different levels of advisories that would appear. Depending on the situation of emergency, administrator users can choose each icon accordingly. After selecting the place to post, the text fields need to be filled for both notification and advisory purpose. Then all the information would be stored in a separate table in the web server database for public users to access. The table view on the left lists all the places have been posted with the created time. Each of the content can be deleted by using the swipe gesture. The data would also be update after the editing the table's content.

8. Share Location View Controller

This view controller (Figure 4.17) controls the switch of sharing or not sharing a user's location with status. It is the users' option that if they want to share their locations in a situation of emergency. However, municipal government would recommend the users to turn on their location sharing function with an initial status by sending a push notification to all the users when an event comes, because it would help the municipal government to track the current situation and make a better decision on how to react. The process of collecting the users' location and status information is by parsing the users' current location to the web server, then after a short period of time, the application would automatically check the users' new location and update it in the database in the web server. After that, administrator users would track the latest users' location and status information.

4.3.2 iOS Device Testing

In Xcode, it provides a simulator is provided that the developer can run and test the application on it. It handles most of the testing requirements even like multi-touch and gestures (swipe, pinch, and tap). However, the ultimate goal of this research is to develop a product that can be ran on each user's device, so a real device testing is necessary through the entire period of application development. The other reason why real device testing is required is that some functions in this application need to call the GPS information every time when the users' location changes, so it would be more convenient to carry the real device around for the location testing.

In summary, this chapter discussed the process of software development and how community members are engaged in climate change adaptation by using this application. The entire discussion of Community Adaptation Mobile Application Design in section 4.2 is based on the system's analysis has done in previous chapter. It takes most of the time to transform community's adaptation needs into system's requirements. In order to provide community users a helpful adaptation tool, the functions of what does this application can do are defined to meet the system's requirements. After that, 4.2.2 Data Management talked about what platform is used for information storage and how does it manage the information. Then, in 4.2.3 User Interface Design section, the designing of user interface considers both community users and municipal controller's different needs. As an important participant in climate change, the user interface design for community users are separated into two parts, which are receiving information and sharing information. For municipal controller user, it has all the features and function community users have and in addition, it is also design for the convenience of posting notification and advisory information. At the end, 4.3 Validity and Verification section makes sure that each function can be realized and running properly on a real device. The code used for developing this application is listed in Appendix E.

In the next chapter, in order to proof the application developed above can handle the real problem, two different simulation plans are prepared to simulate the communication and interaction between community users and municipal controller during an emergency

situation by using this application. It is also a good chance to double check application's performance in the real time.

5 Simulation

Following the system design, application development, and testing, an appropriate simulation plan is needed before passing the mobile application to the users of the system. This chapter presents the simulation of the important function of sharing locations and conditions toward verification of the mobile application. The simulation model presented below tests multiple users' input information and examines different emergency situations. Two simulations are discussed: (5.1) Community-wide Simulation and (5.2) Community Specific Emergency Simulation.

In order to simulate the real time communication between municipal users and public users, another application needs to be developed as the simulator. The main purpose of this application is to simulate the location sharing function, and build up community members as the simulated population. The process of the simulation model is based on different time periods of the emergency event, e.g., before, during, and after the event (as discussed in Section 3.1.4). The detailed processes of these two simulations are described in further detail in the sections below.

5.1 Community-Wide Simulation

For the community-wide simulation, the whole community of Richmond County is under threat by a coming severe weather event. The simulation models how people who live in different areas of the whole county response to threat. This simulation examines how the municipal controller responds to the input of status information from community users, and how local community members may assist each other by sharing their personal status information. The simulation processes for (1) before, (2) during, and (3) after the event are declared below.

(1) Before

The main purpose of the mobile application is to have the municipal government controller initially use the push notification function to post advisory information to the

whole community regarding an emergency weather event that is pending. There are several ways that the municipal government controller can send notifications: e.g., using the local cable TV channels, and/or through a radio broadcast. Alternatively, the municipal controller can use the push notification function of the mobile application (section 4.1.1.4). Community users who have installed the mobile application on their device (iPhone or iPad with application installed) would receive a notification that triggers an alert system default sound. The content of the message would be a short description of the current situation and a request to turn on their location and status sharing function.

Simulation Step 1: 9:00 am, Day 1

Push Notification #1: *Attention please! A severe storm is predicted to make landfall at Richmond County in the next 12 hours by 8:15pm local time. All coastal areas are advised to prepare for strong, gusting winds and intermittent heavy rainfall. All Schools in the county will be shut down before noon today.*

After receiving the notification, users turn on their location sharing function and indicated their current status. As noted in Figure 5.1 below, the municipal controller sees the initial locations and status of 29 users in the system.

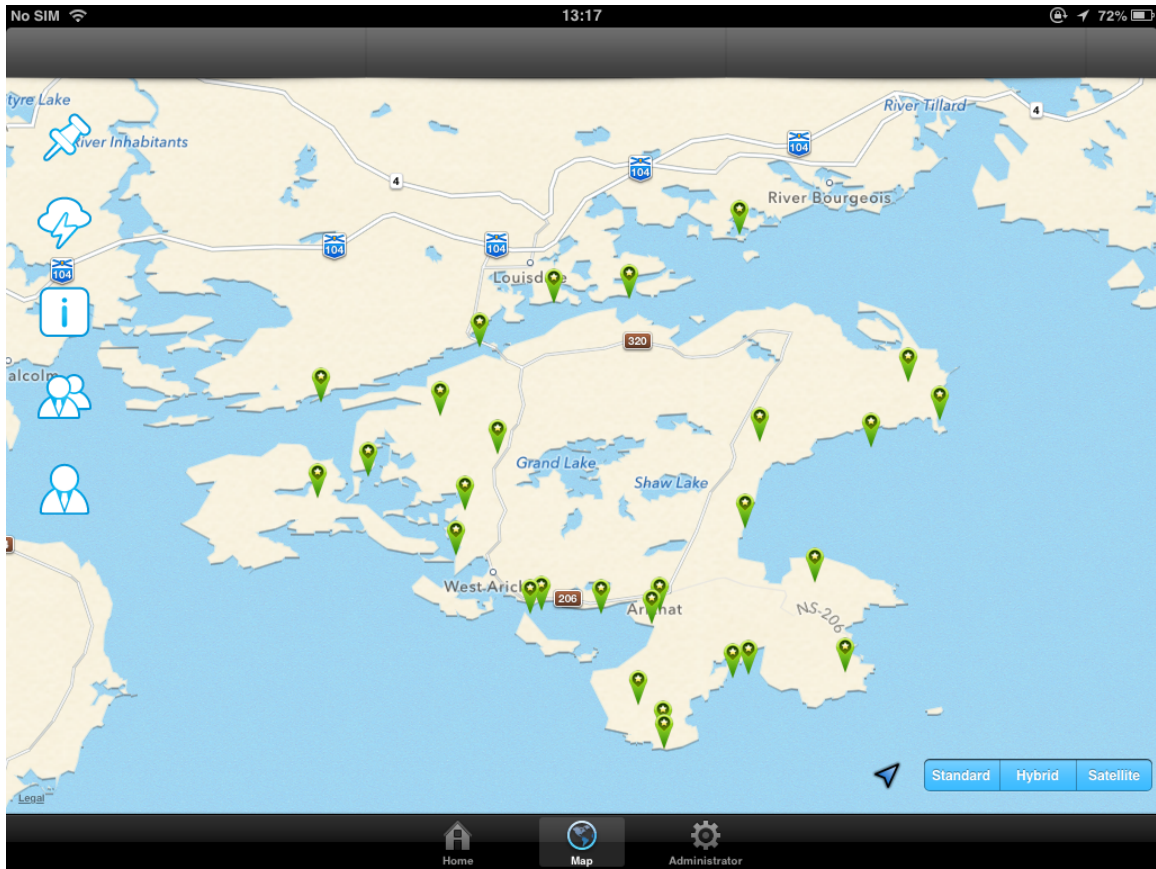


Figure 5.1: Simulation Initial Location and Status Indicator for 29 Richmond County Users

Simulation Step 2: 12:00 pm, Day 1

Push Notification #2: *The storm front is continuing toward landfall in Richmond County. Landfall is predicted in approximately 9 hours at 8:30pm local time.. Be prepared and make sure that there is enough water and food in case of emergency.*

Figure 5.2 below notes that additional local users' information has been received.

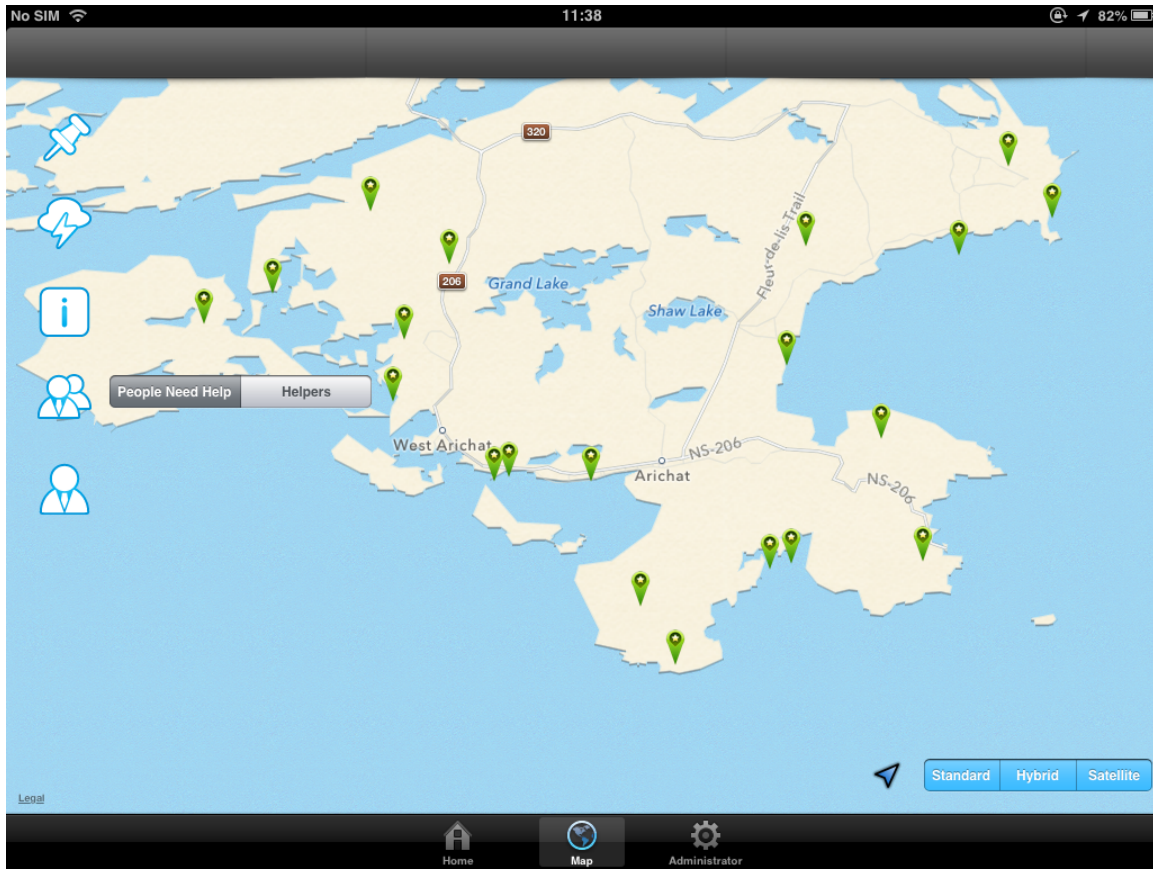


Figure 5.2: Receiving More Information From Community Users

Simulation Step 3: 6:00 pm, first day

Push Notification #3: *Storm now expected to make landfall in approximately 3 hours at 9:00pm local time. Wind speed is 40km/h and rainfall is heavy. Schools and public transportation services have been shut down and will remain closed for Day 2.*

More users have now turned on location sharing and initiated their status as “Ok” (green).

(2) During

Simulation Step 4: 9:00 pm, Day 1

Push Notification #4: *Storm has made landfall. Wind speed is 68km/h. Rains are heavy reducing visibility to zero. Dangerous areas have been identified. Check the advisory map and stay indoors and away from the danger zones as indicated.*

As indicated in Figure 5.3, some users have changed their status from “Ok” (green) to “Concern” (orange) which indicates they are experiencing personal problems attributed to the storm event. At the same time, municipal government controllers are receiving calls and reports about washed out roads and localized flooding throughout the affected areas.

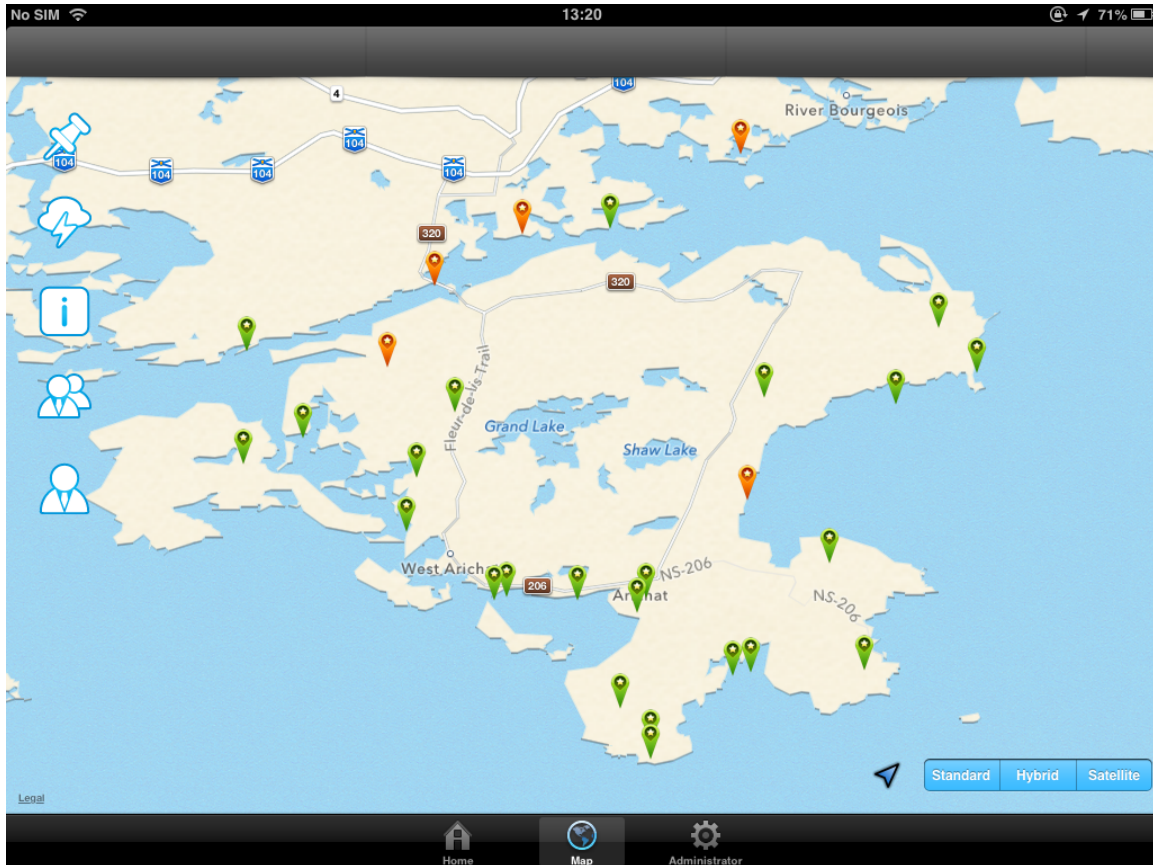


Figure 5.3: Status Changing in Community Users

Simulation Step 5: 12:00 am, Day 2

Push Notification #5: *Check on your neighbor’s status. If you find someone who lives around you and needs help, please ensure your own safety before providing assistance.*

Some community members and application users have received assistance from other users who live near them and can provide help. The community members who are under assistance have changed their status to “Under Help” using the mobile application as indicated on the map (blue status) in Figure 5.4. Information on storm status is being

received (e.g., photos, text information from community observers, updated status of members).

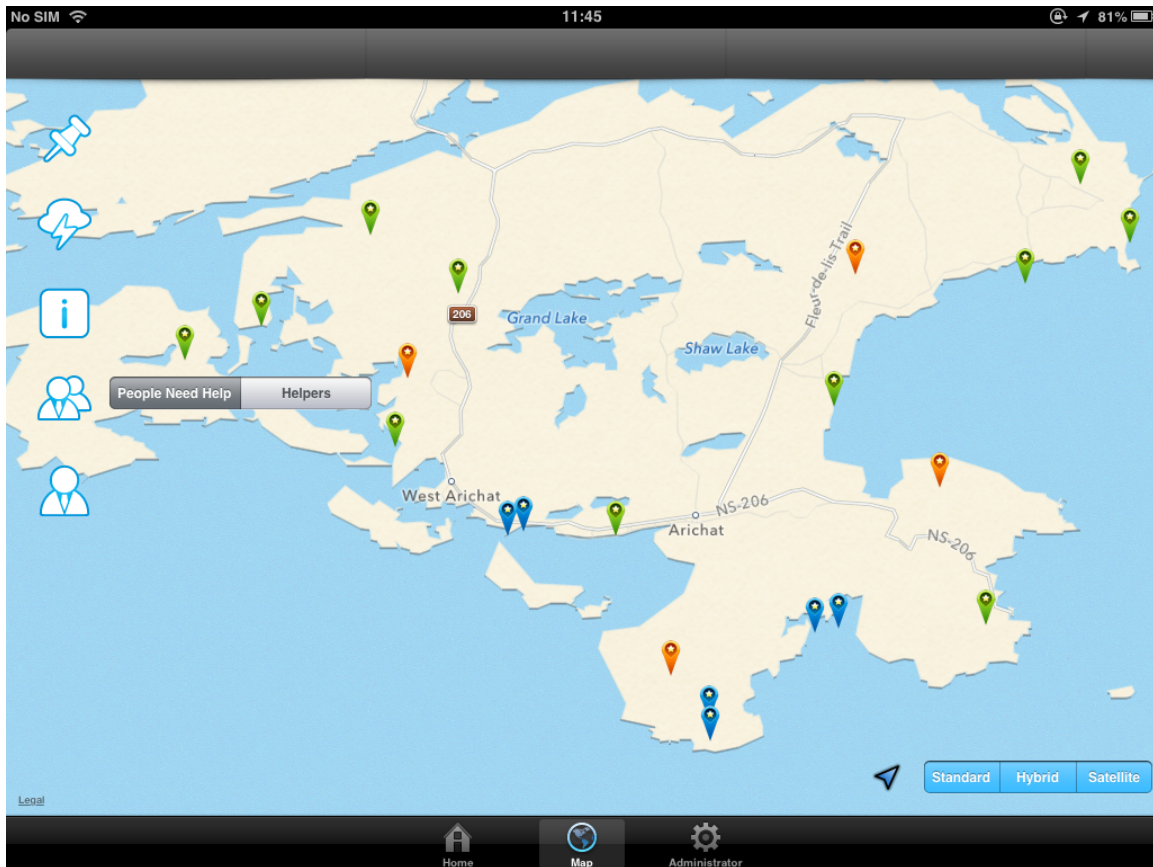


Figure 5.4: Changing Status into Under Help

Simulation Step 6: 3:00 pm, Day 2

Push Notification #6: *Consult the map for designated dangerous areas, and avoid entering the advisory areas near you. If you need to go out, please ensure someone else knows where you are going and can maintain contact with you.*

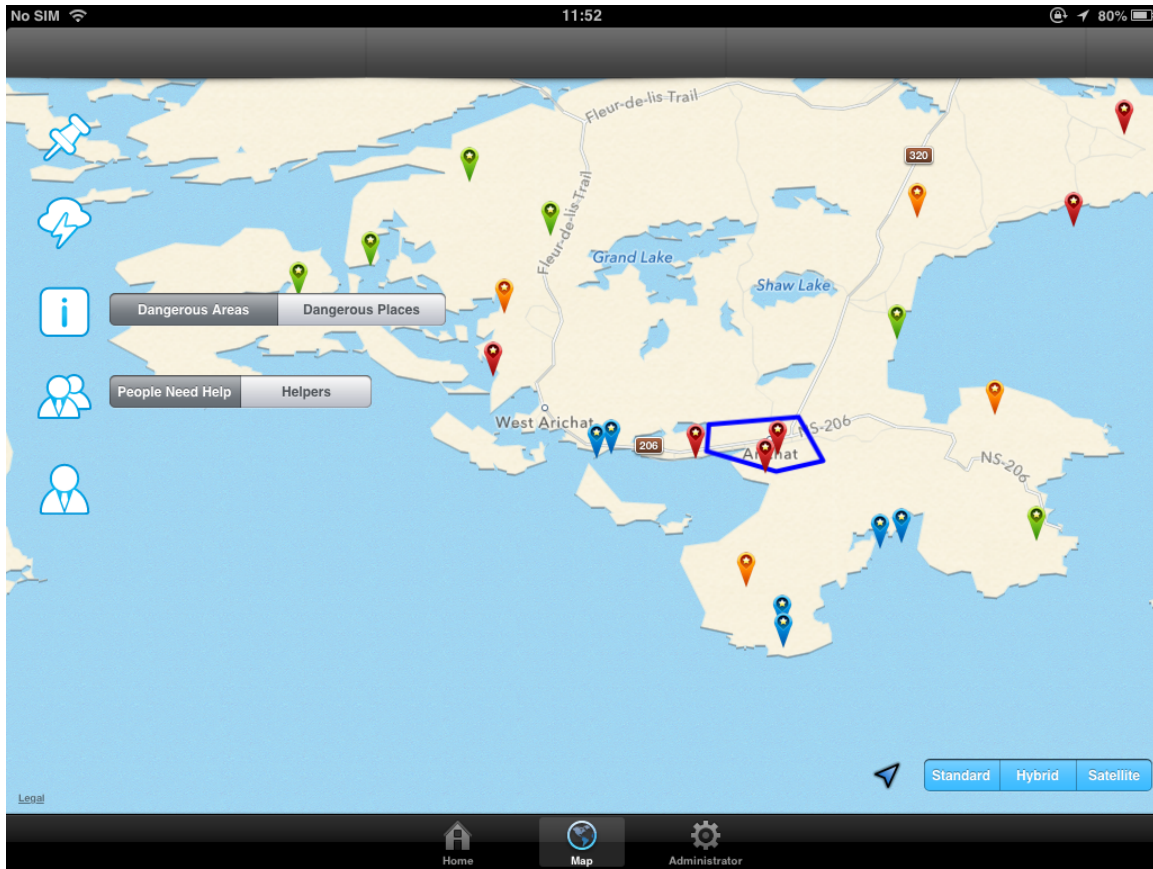


Figure 5.5: Posting Advisory Area Information

Simulation Step 7: 6:00 pm, Day 2

Push Notification #7: *Storm update - Wind speed has reduced to 45km/h, please stay away from dangerous places and areas until further notice. Updates will be released within the next few hours.*

According to the emergency reports and status shared by users, first responders are now in control of the situation and are providing help to people who are still indicating they are in trouble.

(3) After

Simulation Step 8: 9:00 pm, Day 2

Push Notification #8: *Storm has passed through and the weather is clearing. Localized flooding and wind damage means that dangerous places and areas designations remain in place. Check the map regularly for information updates.*

As noted in Figure 5.6, some users have changed their status to “Ok” (green) and some are still under “Help” (red). Information now being received on flooding, wind damage to buildings as residents begin to survey the impacts near them. This information is vetted by the municipal controller, recorded, and passed along to application users.

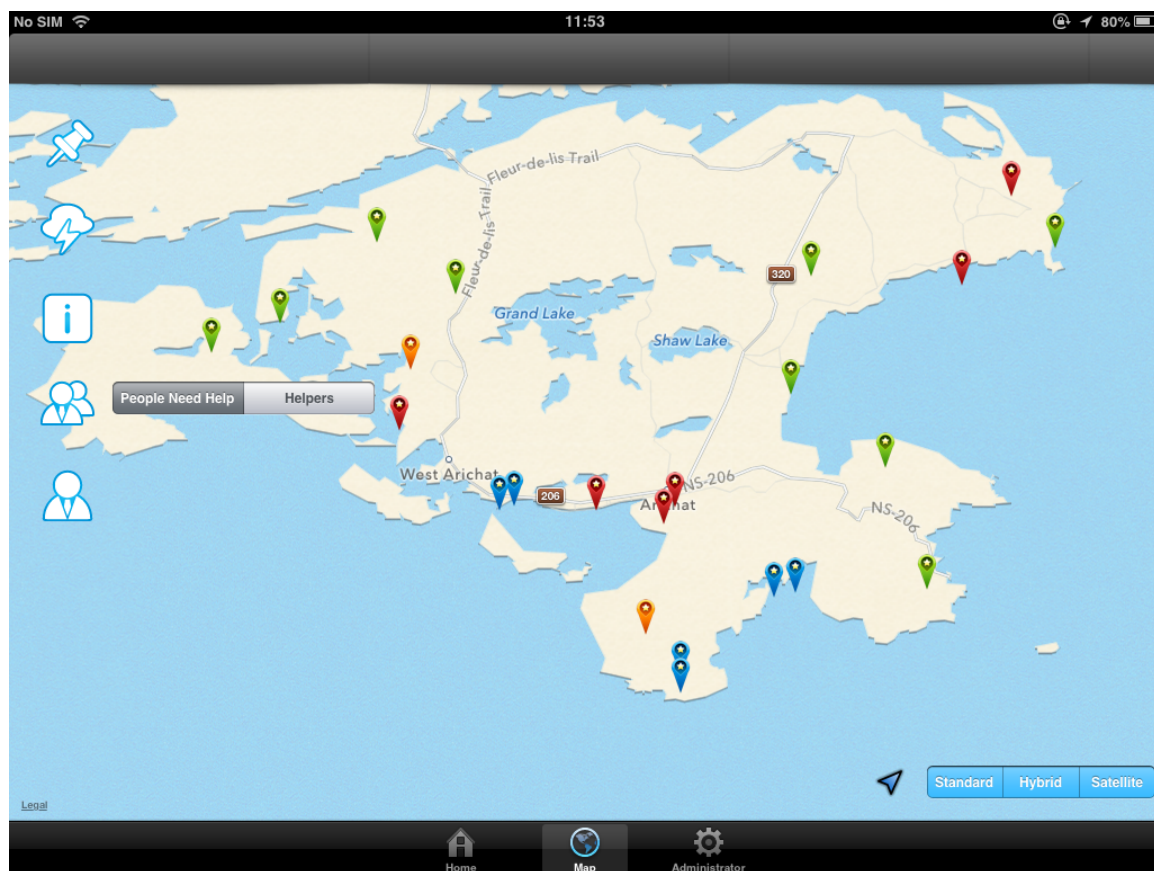


Figure 5.6: Changing Status Back to Normal

Simulation Step 9: 8:00 am, Day 3

Push Notification #9: *New advisory places and areas have been released to the map. Please consult and avoid by selecting alternative travel routes, as needed. Schools are*

reopened this morning and public transportation and school buses have returned to normal outside of the advisory areas.

As noted in Figure 5.7, some users have changed their status from “Concern” (yellow) to “Ok” (green).



Figure 5.7: Community Users Status Update

Simulation Step 10: 11:00 am, Day 3

Push Notification #10: *Storm warnings have now ended. Check your map for area under repair and expect slow downs for reconstruction and repair crews.*

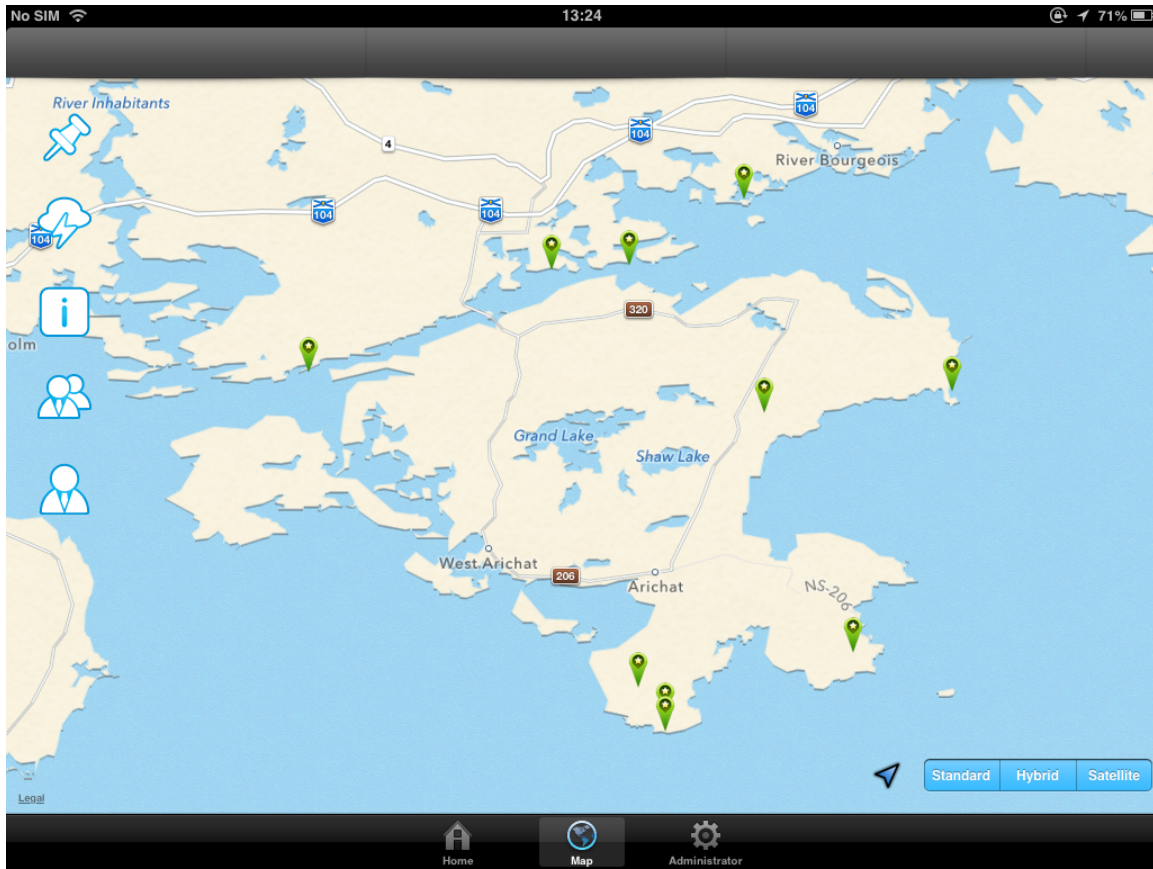


Figure 5.8: Event Finished

This community-wide storm simulation exhibits how the mobile application can be used to (i) communicate with community members; and (ii) permit community members to pass along information and to engage them in assistance as needed. The application also permits further direct intervention as needed, e.g., the dispatch of emergency services ambulances and paramedics to affected areas based on the local feedback of residents that may otherwise not be immediately acted upon. The simulation also illustrates the potential use of the application for non-emergency community use, e.g., the application can be invoked for the case of localized household accidents that require assistance.

The second simulation pertains to a more localized emergency situation that focuses on a designated problem area.

5.2 Community Area Simulation

The small community of Little Anse on Isle Madame has experienced critical isolation conditions when the only access road is flooded by storm water from powerful storm surge (Mostofi, 2011). In such circumstances, people who live in Little Anse are not be able to leave the community and seek immediate assistance from outside, including access to medical help, or even the basics of food and clean water. This community has experienced this problem for decades and recently, the reoccurrence of the washed out road has been more frequent. As one resident has put it “we have been living with this situation for over 30 years”. The difference between this problem and the previous community-wide simulation is that a severe weather event may happen that closes the Little Anse road without any forecast or advanced notification, and yet cause great distress to the community. The steps of this simulation of the case of Little Anse road closure are presented below.

Simulation Step 1: 2:00pm Day 1

A severe seasonal storm landed on the southeast side of Isle Madame associated with a high level of precipitation, high winds, and high tides and failure of the breakwater, which has caused the closure of the Little Anse’s road at the wharf (Figure 5.9). At the same time, the municipal government controller has received status information from the community’s residents indicating the distress they are experiencing.

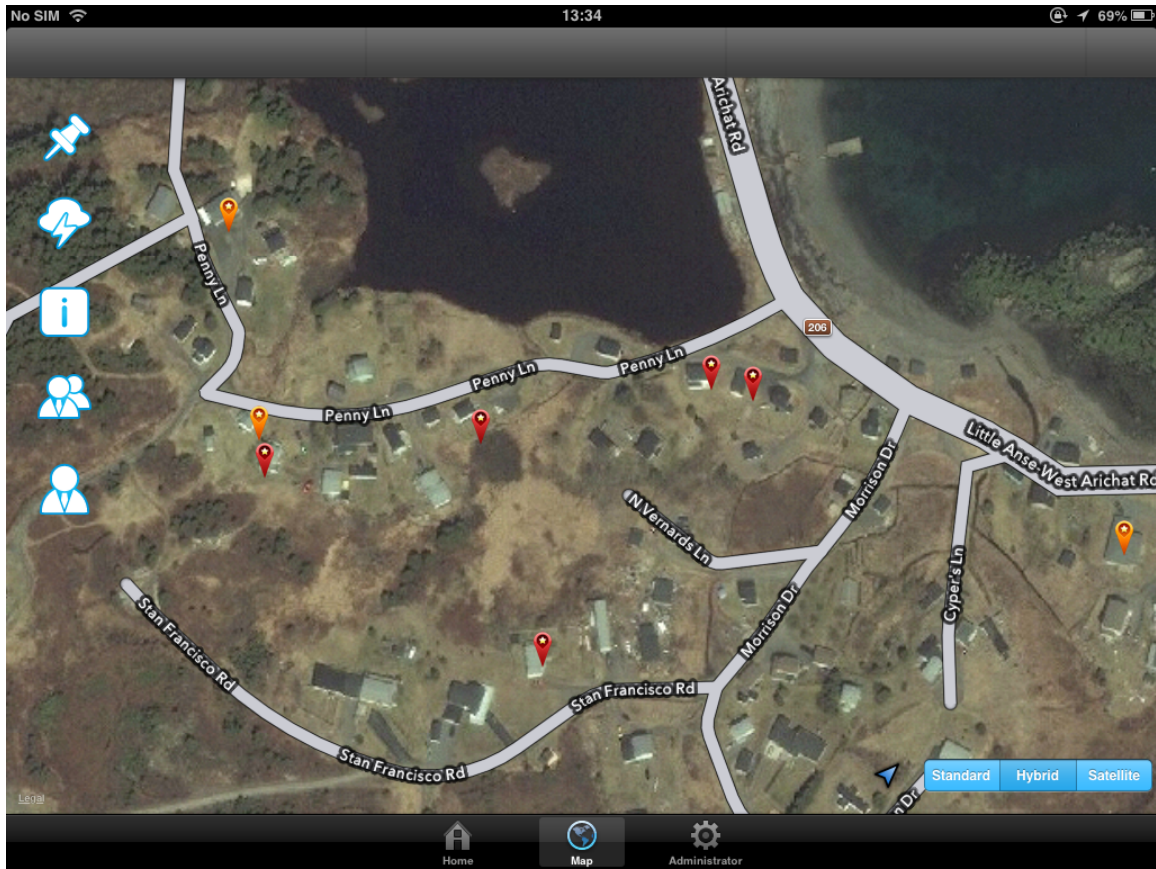


Figure 5.9: Little Anse Road Closure event (top left in photo). (Source: Apple Map)

Simulation Step 2: 2:30 pm Day 1

Push Notification #1: *The Little Anse Road has been flooded and is un-passable. Do not attempt to negotiate this road until further notice.*

The municipal government has dispatched a team of EHS emergency response paramedics from the clinic in Arichat to help the people in Little Anse, however, the road prevents paramedics from reaching community members. Some mobile application users have changed their status to “Under Help” (blue), while others are in the status of “Need Help” (red), as noted in Figure 5.10.

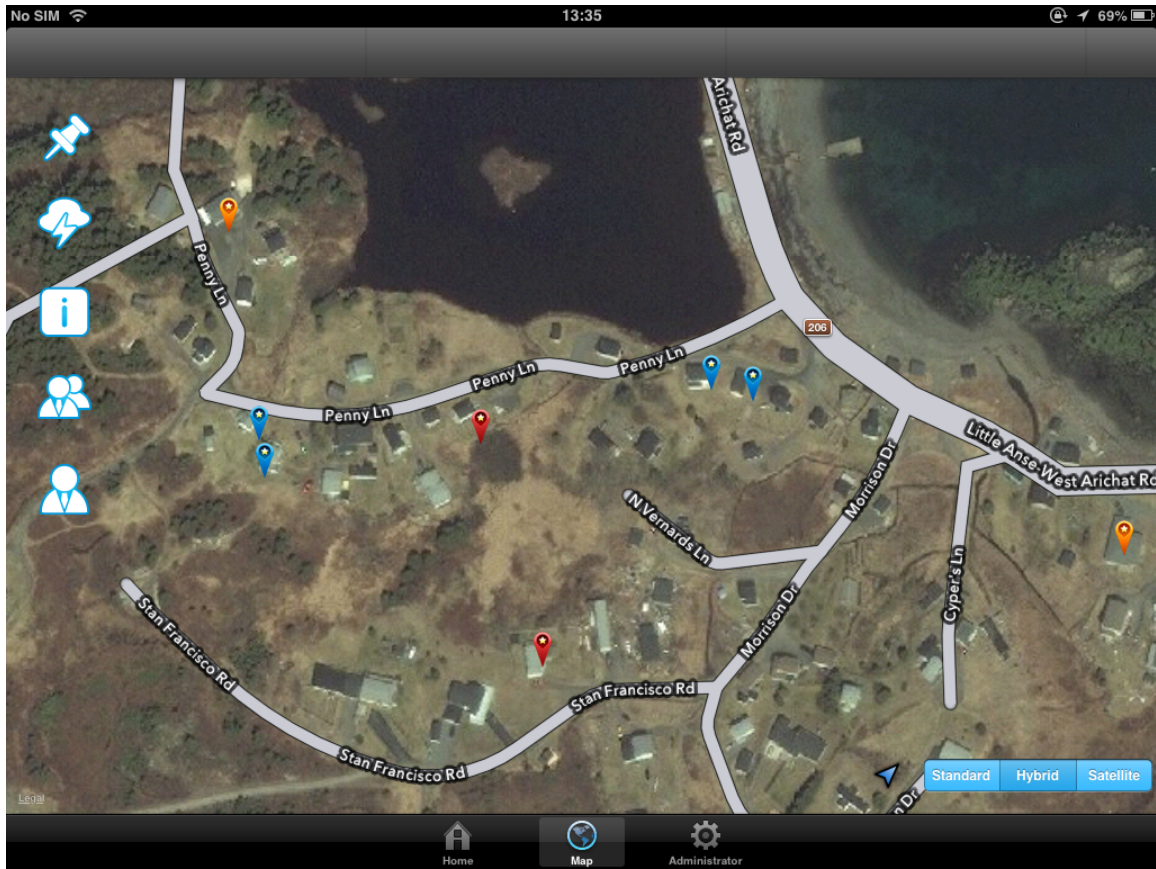


Figure 5.10: Community Users Help Status at Step 2. (Source: Apple Maps)

Simulation Step 3: 3:00 pm Day 1

Some users in Little Anse have shared their location and indicated that they can provide local assistance as “Helpers”. In the event that the Emergency response team of paramedics is unable to reach people in distress, Helpers are able to relay calls and obtain advice from the paramedics off-site. Meanwhile, other users have changed their status from “Need Help” to “Under Help”, which means that they are receiving assistance from the “Helpers” as people who live around them in Little Anse (Figure 5.11).

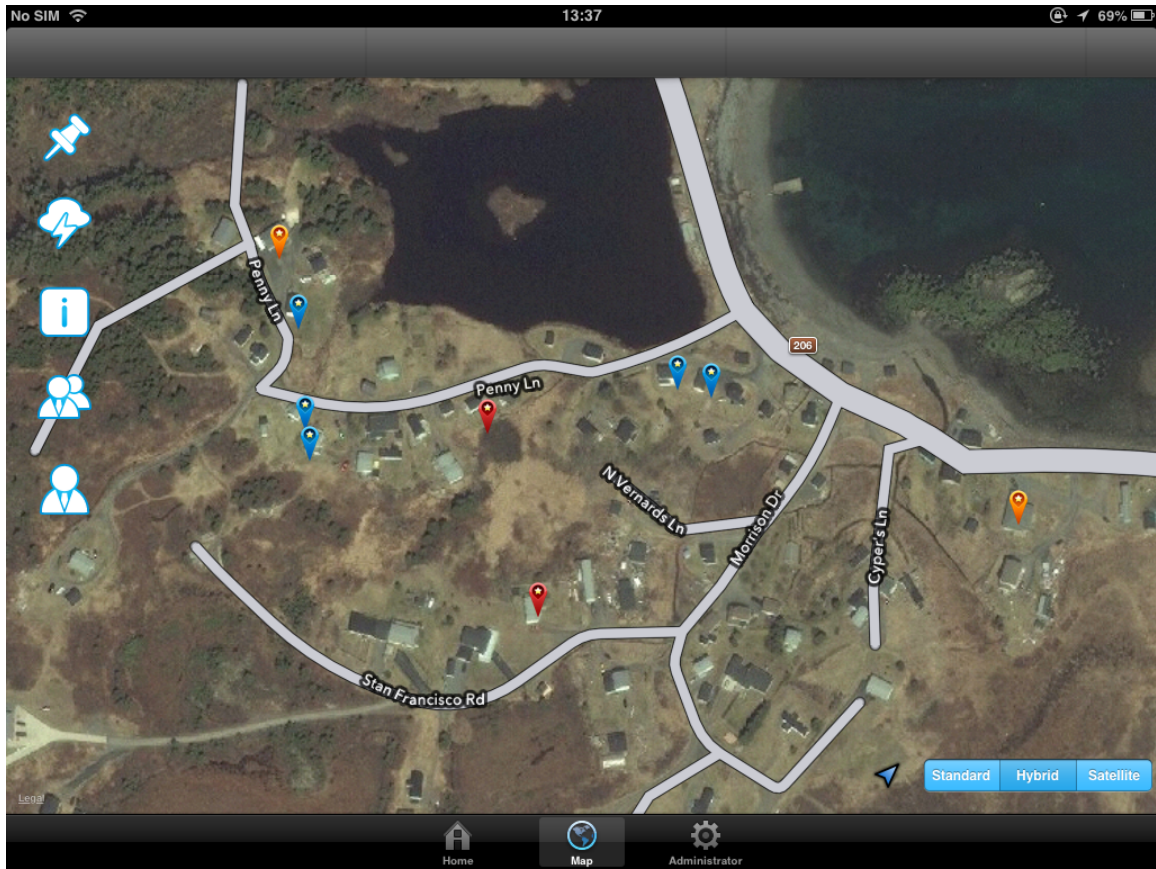


Figure 5.11: More Community Users Receiving Help from “Helpers” at Step 3. (Source: Apple Map)

Simulation Step 4: 5:00 pm Day 1

Push Notification #2: *If you are currently in Little Anse and can provide assistance to those around you in need, please check the map, identify yourself and inquire if you can be of assistance. For medical information, contact the paramedics immediately for advice.*

Three hours after the flooding and road washout, people of Little Anse have received assistance, as noted in the status graphic of Figure 5.12.

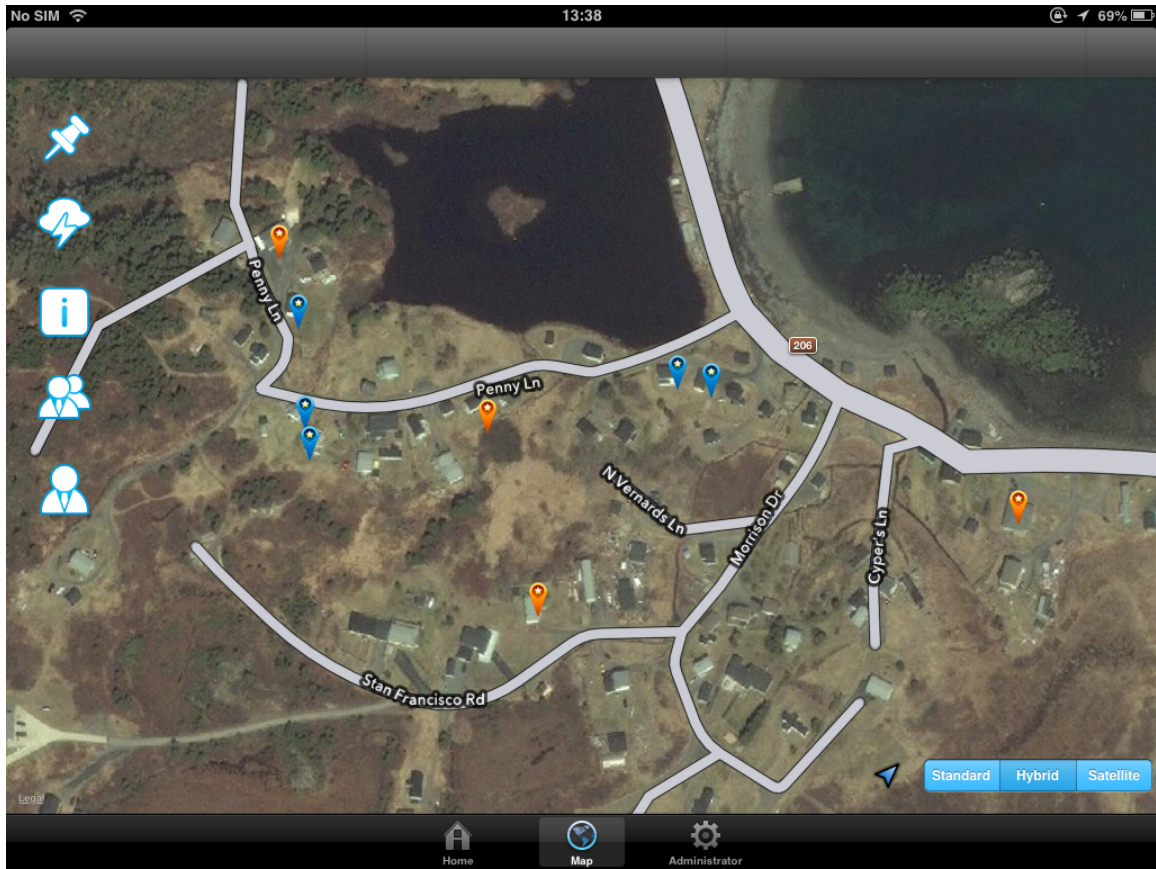


Figure 5.12: Most Community Users Are Out of Trouble (Source: Apple Map)

Simulation Step 5: 6:00 pm Day 1

Four hours after the floods and road washout to Little Anse. Some users have changed their status from “Under Help” to “OK”, which signals the end of the emergency situation (Figure 5.13).

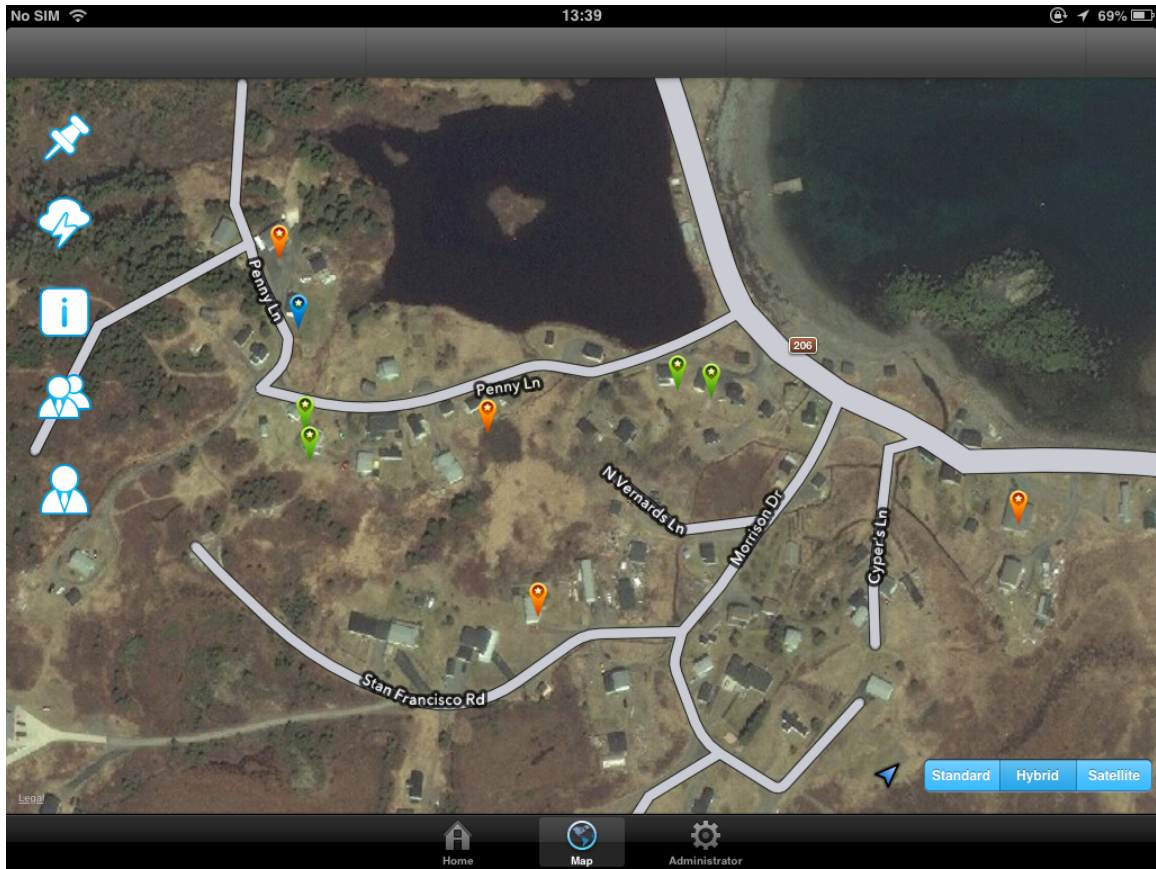


Figure 5.13: Event Completed (Source: Apple Maps)

In Summary, according to the performance of the above simulation, functions defined and implemented for this adaptation mobile application work properly. This means that this application can handle multiple users information input and output, which requires high performance of data exchanging between users and server.

6 Conclusions

Developing the adaptation and action framework for coastal communities defines the community profile, current threats and vulnerability, the construction of an action planning process for emergency response, and the development of the mobile application and adaptation website for action plan implementation. These presented steps also accomplish the thesis objectives proposed in Section 1.3 above.

Previously, the research questions asked in section 1.2 are

- (1) How can the current status and vulnerability of a coastal community be described?
- (2) What are the elements of an effective community adaptation plan?
- (3) What are the processes required for operational action plan implementation?

Developing the climate change adaptation framework helps coastal community understand, collect, and prepare adaptation situation. This is outlined in section 3.1 and it meets the requirement for question (1). Then in section 3.2 System Analysis section, two effective adaption solutions were outlined, adaptation of mobile application and community adaptation website. Lastly, in chapter 4, implementation was introduced. It described whole process of how to transform adaptation needs into a real application.

Delivering adaptation and action planning application to local community is the main outcome of this research study. By using the developed mobile application, the municipal government and the local community are engaged as participants in effective community action and emergency response. The detailed achievements of this research are summarized in the next section.

6.1 Summary

As this research was nearing completion, a large flood occurred in mid June 2013 in Calgary as well as in other sites in Alberta. This event is the largest flood that has happened in Alberta since the last 100 years. It caused over \$1billion CAD of damages,

and at least 3 fatalities as a direct consequence of the flood (CBC Toronto, 2013). It was unpredictable and so sudden, that people did not even have time to prepare. Although local and federal government have provided some initial help, it cannot extend to each individual requiring assistance over an extended period of time.

There is one thing in this event we can learn and also can answer this question. Some of the people affected by the flood got interviewed and were asked about the current situation and the availability of assistance. Most responded that they got help from their neighborhoods or someone unaffected by the flood. One woman in an interview was going around and giving away hotdogs, because she had hotdog that can be shared with others. Some other people even offered to provide accommodation for the people displaced (CBC Podcast Interview, 2013). People helping people in the community is a good exercise in climate change adaptation and action planning. In this research, one of the results is an iPad application, which has a function that allows people to share their location and status. Like the situation above, if the people in need and the people who can offer accommodation can communicate through a mobile application, it would improve the effectiveness and efficiency of the emergency response. Future studies can focus on specific areas, which would help people in different situations.

The development of adaptation framework in this thesis is based on the review and summary of ICLEI, MCCAP adaptation framework and the experiment in Appendix B, C, and D. The purpose of developing this framework is to simplify the complex process of identifying climate change impacts. For this research, the scope of the adaptation framework is defined for the purpose of action planning and implementation. As the first component of this framework, the development of community profile is based on the collection of community's typical deposits of information according to the four pillars. This location information is used as a part of implementation and community has the responsibility to customize and update it in the system. In addition, communities can follow other processes in this framework to collect information and identify adaptation requirements. The idea of developing an adaptation framework is to provide coastal communities with a starting point, which allows them to put adaptation into action.

By the time coastal communities start preparing adaptation and action planning, some information should be constantly updated because of new requirements that may also come up. To solve this problem, an information system development process is defined to meet another objective in this research, which would help coastal communities to transfer adaptation needs into information technology. Most of the time, the research of climate change impacts or study is done at once and hard to keep tracking the results and outcomes continuously. In this research, the work of transforming adaptation needs into an information system is done in previous chapter and identified the system's requirements.

After framework development and system analysis phase, nine main functions are defined for the implementation of the system according to the system's requirement and adaptation needs. The advantage of iOS described in previous chapter makes it the best device to implement and run this climate change adaptation system. By the time of the development of the iPad application, each function was developed and tested on an iOS device. The data exchange and storage in this application is based on an online server called Parse, which is a PaaS (Platform as a Service) cloud service. It provides professional cloud services and powerful APIs, which allows the developers to create high quality applications. Uploading and downloading data from this server is functional and effective, which has been tested during the run time. All the functions defined in the system requirements have been proven to be working. The other important part of the implementation is to develop an adaptation website to support this application and provide adaptation policy, local news, and events update.

At last, this research ends up with a two different scales of simulation, community wide and small area wide. The concept of this simulation is based on several scenarios during the different periods of an event. Because of the limited number of devices this research was tested on, it is not realistic to simulate hundreds of users by using the same amount of devices. In order to simulate multiple users, a simple simulation application was developed by using a single location sharing function, which allows users to share and change their location and status information. This simulation application is a collection of

multiple users and each user is independent from each other. The results of the simulation have proven that this adaptation application allows multiple users to share and check information at the same time.

At the beginning of this research, the question been asked was, “How can coastal communities clearly identify climate change impacts, how to adapt and how to implement?” The results of this research provide one of the possible answers to this question. The impacts of climate change are complex and severe, which involves not only environmental scientist but also policy makers. People who work in other areas may have different ideas or understanding about how to solve these problems. To get more people get involved in climate change adaptation, one important strategy is to increase public awareness and break down the boundaries between environmental science and other sciences, so we can get a chance to see it from different perspective. It is also the governments and schools’ responsibility to educate the young children about climate change, especially how to respond or protect themselves when events happen.

6.2 Future Study

Since this research is done using adaptation mobile application and website development, some questions may be raised about the concern of accessibility and privacy. In the next sections, these questions would be discussed accordingly. As an extension to this research, bringing in more new functions based on the system’s requirements or using this application for both research and commercial use could be explored. In the following sections, the recommendations for the future study would also be discussed.

6.2.1 Questions and Concerns

According to the result of system analysis, the implementation of climate change adaptation and action planning is based on the widely used information technology. The most frequent asked questions for this adaptation solution is that what if there is no Internet connection, what if the power is down, and privacy issues. These are good questions to ask, because it reminds us to think about each possibility that may occur,

because anything could happen during a severe event caused by climate change. Any failure of response or reaction would cost immense loss. To answer these questions, some point and thoughts are discussed as followed.

First, by implementing climate change adaptation and action planning, technologies are not the only solution to solve all the adaptation problems. There might be other methods or solutions that communities can use. However, each of these methods or solutions does not work independently. Depending on different situations, communities should learn to incorporate different adaptation needs and requirements.

Second, it is the fact that any damage to Internet infrastructures or facilities would affect network connections, which would cause adaptation mobile application and website to be dysfunctional. However, this would come from another research area about improving the reliability of the networking environment, which allows people to access Internet uninterrupted. Also, providing other kinds of network connections similar to 3G/4G or satellite would also be options.

Third, power outage is another concern in the implementation of adaptation application and website. It is obvious that electronic devices like routers and modems, which are important components of Internet connection, cannot work properly without power. For some other devices like cellphone, tablet, or laptop, battery can afford them several hours of operation. So, the biggest concern here is if the power system can get back to normal when there are some damages on the power grid and how long it takes to recover. For some communities, people have their own generator in their house in case of power outages. Municipal government as well, would also have their backup power system, which can be used in a situation of emergency. Any power recover or backup technologies would provide a better chance of getting network connection back to normal.

At last, like any other social network media, Facebook, Twitter, privacy issues are big concerns. As an important feature of this adaptation application, location and status

sharing could provide communities a good tool to response and react, however, how to keep users privacy through the sharing process is important as well. As a solution for privacy concerns, before users start to share their location and status information, the application would give them an option that either they can share the information to all users, or they can only share it with administrative users. Users only share their information when they think it is necessary, otherwise, there is an option in the application that allows users to close the sharing function. At the same time, their information would be deleted from the system, so no one can have access to it.

6.2.2 Recommendations

The two main outcome of this research is modeling the adaptation framework to help coastal community reduce the impacts of climate change and by using information technology to implement it. Future studies can do more specific work based on these two parts. Below are some recommendations.

For the adaptation framework, as discussed previously, there is no single solution that can solve the complex problem of climate change. Constant research would be a better option to have a good understanding of community's adaptation needs and implementation difficulties. One recommendation is that communities do not have to exactly follow the steps in this adaptation framework, if they have their specific problems to deal with. Some communities may focus on profile development, maybe because they did not do enough on collecting those information or things changed a lot from past few years. Some other communities may want to emphasize on implementation because they already have a better understanding of the impacts of climate change and just want to find a way to reduce the impacts. So, communities can base on their own adaptation needs to prepare a framework.

The other important outcome of this research is the development of adaptation mobile application and website. For the application, it is based on the analysis of adaptation needs and action requirements, which would allow the people who live in the community

to actually participate. There are several areas people can do further work on this application, which are developing a similar application for other platforms like Android or Windows, extending the functions for their own adaptation needs, and extracting the data generated from the system and analyze it for research. For the website, communities can do some advanced development such as providing the same location based advisory information on the main page, developing adaptation schools on the website, and provide technical support for the mobile application.

7 Bibliography

- Adger, Neil. 2005. *Approaches to Vulnerability to Climate Change*. Centre for Social and Economic Research on the Global Environment. University of East Anglia and University College London. ISSN 0967-8875
- Adger, Neil. Brooks, Nick. 2004. *New Indicators of Vulnerability and Adaptive Capacity*. Tyndall Centre for Climate Change Research. University of East Anglia. p.122
- Adger, Neil. *Vulnerability*. 2006. Tyndall Centre for Climate Change Research. School of Environmental Sciences. Global Environmental Change 16 (2006) 268–28
- Australian. 2005. *Climate Change Risk and Vulnerability*. Australian Greenhouse Office. ISBN: 1 920840 94X
- Bizikova, Livia, Neale, Tina and Burton, Ian. 2008. *Canadian Communities' Guidebook for Adaptation to Climate Change*. First Edition. Environment Canada and University of British Columbia, Vancouver. ISBN: 978-1-100-10839-1
- Boudreau, Aleasha and Theriault, Michelle. 2012. *Isle Madame Research Project: Isle Madame Historical Documentation and Storm Monitoring Project*. Université Sainte Anne, Marine Research Centre. p.65
- Canada. 2006. *Agreement on The Transfer of Federal Gas Tax Revenues*. Governments of Newfoundland and Labrador and Canada.
<http://www.ma.gov.nl.ca/ma/publications/gta/agreement.pdf> Accessed: 2012/06/10
- Carpenter, Chad. 2007. *The Bali Action Plan: Key Issues in The Climate Negotiations*. In UNDP. *The Bali Road Map: Key issues under negotiation*. United Nations Development Program Environment & Energy Group. pp.6-21.
- Dangermond, Jack and Artz, Matt. 2010. *Climate Change is a Geographic Problem: The Geographic Approach to Climate Change*. ESRI. United States of America.
<http://www.esri.com/library/ebooks/climate-change.pdf> Accessed: 2012/09/02
- EMO. 2011. *R.M. OF LUMSDEN #189*. Emergency Measures Organization. P.74
- FCM. 2009. *Five-Milestone Framework for Reducing Greenhouse Gas Emission*. Green Municipal Fund. Toronto: Partners for Climate Protection. p.18
- FEMA. 2010. *Developing and Maintaining Emergency Operations Plans*. Federal Emergency Management Agency. P.124
- Fisher, Graham. 2011. *Municipal Climate Change Action Guidebook*. Service Nova Scotia and Municipal relations. p.36

- Gupta, Sonia. *Environmental Law and Policy: Climate Change as a Threat to International Peace and Security*. 2007.
<http://www.perspectivesonglobalissues.com/0401/envirolaw.pdf> Accessed: 2012/09/04
- Hart, Maxwell. 2011. *Geographic Information Systems and System Dynamics-Modelling the Impacts of Storm Damage on Coastal Community*. Telfer School of Management. University of Ottawa. p.233
- Homeland Security. 2008. *National Incident Management System*. U.S. Department of Homeland Security. Washington. <http://www.nwcg.gov/branches/pre/nrf-nims/nims.pdf> Accessed: 2012/09/06
- Homeland Security. 2012. *Incident Action Planning Guide*. U.S. Department of Homeland Security. Federal Emergency Management Agency. Washington, DC. [http://www.uscg.mil/hq/cg5/cg534/nsarc/FEMA%20Incident%20Action%20Planning%20Guide%20\(IAP\).pdf](http://www.uscg.mil/hq/cg5/cg534/nsarc/FEMA%20Incident%20Action%20Planning%20Guide%20(IAP).pdf) Accessed: 2012/09/05
- ICLEI 2012. *Changing Climate, Changing Communities: Guide and Workbook for Municipal Climate Change Adaptation*. Local Governments for Sustainability. <http://www.iclei.org/index.php?id=10832> Accessed: 2012/06/03
- Lacucci, Anahi. 2010. *A Step-By-Step Guide on How to Use Ushahidi Platform*. Ushahidi. <http://community.ushahidi.com/uploads/documents/Ushahidi-Manual.pdf> Accessed: 2013/05/05
- Lemmen, D.S., Warren, F.J., Lacroix, J., and Bush, E., editors (2008): *From Impacts to Adaptation: Canada in a Changing Climate 2007*; Government of Canada, Ottawa, ON, 448 p.
- Litman, Todd. 2012. *Planning Principles and Practice*. Victoria Transportation Policy Institute. P.34
- Neumann, James and Price, Jason. 2009. *Adapting to Climate Change: The Public Policy Response: Public Infrastructure*. RFF Report. Washington, DC: Resources for the Future. p.41
- Newfoundland and Labrador. 2009. *Integrated Community Sustainability Plan Framework*. Department of Municipal Affairs. <http://www.ma.gov.nl.ca/ma/publications/icsp/newfoundland-and-labrador-icsp-framework.pdf> Accessed: 2012/06/10
- Nova Scotia. 2011. *Nova Scotia 2010-2011 Statement of Mandate*. Emergency Management Office. P.17
- Pakdel, Sahar. 2011. *Spatial-Temporal Modeling for Estimating Impacts of Storm Surge*

and Sea Level Rise on Coastal Communities: The Case of Isle Madame in Cape Breton, Nova Scotia, Canada. Telfer School of Management. University of Ottawa. p.250

Park, Chad. Purcell, Mike. Purkis, John. 2009. *Integrated Community Sustainability Planning*. The Natural Step Canada. p.127

Patwardhan, Anand. Semenov, Serguei. Schneider, Steve. 2007. *Assessing key vulnerabilities and the risk from climate change*. IPCC WGII Fourth Assessment Report. Cambridge University Press, Cambridge, UK, 779-810.

Richardson, G. 2010. *Adapting to Climate Change: An Introduction for Canadian Municipalities*. Ottawa, Ontario. Natural Recourse Canada, 40 p.

Richmond County. 2009. *Richmond Integrated Community Sustainability Plan*.
http://www.richmondcounty.ca/images/pdf/public_works/Draft%20ICSP%20Sept%202009.pdf. Accessed: 2012/09/02

Shapiro, Janet. 2001. *Action Planning Toolkit*. World Alliance for Citizen Participation.
http://www.aces-tools.or.at/images/doku/action_planning.pdf Accessed: 2012/09/02

Wongbusarakum, Supin and Loper , Christy. *Indicators to Assess Community-Level Social Vulnerability to Climate Change*.
<http://www.socmon.org/download.ashx?docid=64623> Accessed: 2012/08/28

Xie, Lian. Hammerling, Dorit. Liu, Bin, and Fuentes, Montserrat. 2013. *2013 Atlantic Tropical Cyclone Outlook*. North Carolina State University. p.11

Appendices

Appendix A: Charlottetown Green House Gas Emission Inventory Summary

As a member of PCP, the city of Charlottetown has involved the five-milestone program to reduce GHG emission. By using the data provided by the city planner, I prepared this GHG emission inventory, which is the first milestone in this program. This inventory can help local government have a basic idea about the amount of GHG produced each year by municipality and community. Table 1 and 2 presents the summary of emission by sector and source from the municipality. In table 3, 4 and 5, it shows the total GHG emission from resident, commerce and industry in the city of Charlottetown.

Sector	Emission CO ₂ e (t)
Municipal Fleet	1196.39t CO ₂ e
Street Lighting	535.26t CO ₂ e
Buildings	1049.7t CO ₂ e
Water and Sewage	744.61t CO ₂ e
Total	3525.96t CO ₂ e

Table 1: Green House Gas Emission By Sector

Source	Emission CO ₂ e (t)
Electricity	1277.4t CO ₂ e
Diesel	558.22t CO ₂ e
Fuel Oil	985.57t CO ₂ e
Propane	3.53t CO ₂ e
Gasoline	713.85t CO ₂ e
Total Emission	3538.57t CO ₂ e

Table 2: Green House Gas Emission By Source

Source	Energy Use	CO ₂ e Emission
Electricity	104856.164MWh/year	76545t CO ₂ e
Fuel Oil	27710000L/year	72046t CO ₂ e
		Total: 148591t CO ₂ e

Table 3: Resident Green House Gas Emission

Source	Energy Use	CO ₂ e Emission
Electricity	252746.988MWh/year	209780t
		Total: 209780t CO ₂ e

Table 4: Commerce Green House Gas Emission

Source	Energy Use	CO ₂ e Emission
Electricity	515342.466MWh/year	376500t CO ₂ e
Total		376500t CO ₂ e

Table 5: Industry Green House Gas Emission

Appendix B: Comparison of ICLEI and MCCAP Adaptation Framework

As is known, ICLEI and MCCAP's adaptation plan framework contains two parts, which are mitigation and adaptation. The following two tables compare the details of each adaptation process respectively. The comparison of these two adaptation frameworks can help us simply understand the similarity and advantage of each plan.

Mitigation

Steps Milestone	MCCAP	FCM Milestone
1	Energy & Emission Information Objective: Complete emission spreadsheet.	Milestone 1 Create a GHG Emission Inventory and Forecast Objective: (a) Complete GHG inventory (like Kimberley) (b) Making a forecast by using economic and business growth
2	Energy & Emission Inventory Table Objective: Complete inventory table (Summary of Spreadsheet) using Stantec Workbook	
3	Setting Goals and Actions Objective: (a) Identify goals for reducing GHG (category in inventory table) (b) Develop achievable actions (fully realize the goals)	Milestone 2 Set an Emission Reduction Target Objective: (a) A 20 per cent reduction in GHG emissions below baseline levels for municipal operations within 10 years. (PCP recommended) (b) A six per cent reduction in GHG emissions below baseline levels for the community within 10 years. (PCP recommended) Milestone 3 Develop a Local Action Plan Objective: (a) Summary of baseline emissions forecasts and targets (b) A set of existing and proposed emissions reduction actions (c) Implementation strategies, including the resources involved (d) Input from stakeholders Milestone 4 Implement the local action plan or a set of activities Process: (a) Staffing and communications Communication: Communicate the objectives of PCP and the LAP (Local Action Plan) to council.

	Time Frame Goals and Actions should be accompanied by Time Frame. (Short-term 1-2 year, medium-term 3-4 years and long-term 5+ years)	The information must be communicated to the city/town manager and department heads. • Resource: Obtain adequate resources (identify the funding and staff/outside assistance required) • Measures: Ensure the implementation of existing measures (b) Financing and budgeting (c) Project timeline • Prepare an implementation schedule that meets the emissions reduction goal in the target year • Integrate the schedule with existing processes and responsibilities • Provide time for stakeholder review and input (d) Stakeholder engagement: Strong stakeholder engagement is required during the development and implementation phases to ensure support for the plan • Consider establishing a "community task force" • Recruit volunteers and interns to assist with implementation and promotion of the plan to the public
		Milestone 5 Monitoring progress and reporting results Process: (a) Track the results of specific emissions reduction measures (b) Update the inventory (c) Engage stakeholders and decision-makers (d) Report to stakeholders and to FCM (e) Realizing goals

Table 6: Mitigation Comparison

Adaptation

Step Milestone	MCCAP	FCM (ICLEI)
1	Adaptation Team/Committee Objective: (a) Member of team consists government officials, staff and stakeholders (b) Establish a mandate and terms of reference for preparing the plan (c) Identify limits of the committee's authority and level of resourcing	Milestone 1 Initiate (a) Identify stakeholders (b) Build climate change adaptation team (c) Identify an adaptation champion

2	Impacts and Hazards Objective: (a) Deep understanding of the kind of climate change municipality has experienced (b) Look at what climate changes are expected and how these changes may exacerbate or cause new hazards in the municipality, today and future. (c) Identify any opportunities which may result from future changes in the climate	(d) Take a first look at climate change impacts and existing adaptation actions (e) Pass council resolution and community charter Milestone 2: Research (a) Initiate research on climate changes
3	Affected Locations Objective: (a) Identify where the issues may occur (b) Determine which will become an issue that should be addressed through adaptation	(b) Refine impacts and consider service areas for each (c) Vulnerability assessment (study of sensitivity and adaptive capacity)
4	Facilities, Infrastructure and Service Delivery Objective: (a) Identify key facilities and infrastructure in the municipality (b) Identify where they are located and determine if some facilities and infrastructure will be more affected by climate change than others (c) Evaluate the impact of climate change on the delivery of municipal services	
5	Social Consideration Objective: (a) Determine which residents will be most affected (b) Identify which residents, local groups and agencies can help Economic Consideration Objective: (a) Identify how the impact affect the most important sector of the economy in the municipality and the economic goals the municipality has identified in the ICSP (b) Understand how climate change will affect the financial resources of your municipality Environment Consideration Objective: (a) Evaluate how the natural environment will be affected and identify potential environment problems resulting from climate change issues	

	(b) Understand how climate change may help or harm important species or natural areas found within your municipality	
6	Priorities for Action Objective: (a) Prioritize the climate change adaptation issues considered by the committee to pose the greatest risk to the municipality, both now and in the future (b) Identification of potential courses of action for priority adaptation issues (c) Develop an approach to integrate adaptation priorities into municipal planning documents	(d) Risk assessment (consequence and likelihood of impacts) and prioritization Milestone 3: Plan (a) Establish adaptation vision and objectives (b) Set goals (c) Identify options and actions (d) Identify possible drivers and constraints (e) Evaluate actions against drivers and constraints (f) Determine appropriate baseline and indicator data (g) Examine financing and budget (h) Establish implementation schedule (i) Create action plan (j) Launch plan Milestone 4: Implement (a) Begin implementation (b) Solidify support from Council and community (c) Use appropriate implementation tools (d) Follow terms of action plan (e) Report on successes regularly to maintain momentum Milestone 5: Monitor/review (a) Assess new information and review drivers (b) Track implementation progress (c) Evaluate effectiveness of actions using baseline data and indicators (d) Communicate accomplishments (e) Investigate future adaptation options and actions (f) Revise adaptation plan (g) Launch next round of adaptation plan

Table 7: Adaptation Comparison

Appendix C: Charlottetown ICSP Template

The city of Charlottetown has engaged in preparing climate change adaptation plan on their own, which is called Sustainable Community. By using the work they have done on action planning, this appendix develops the ICSP template for the city of Charlottetown.

Pillar 1: Environmental Consideration

Issues	Actions
Water Resource	Protect and restore watersheds by developing watershed management plans for watersheds and reducing/correcting the effects of watercourse siltation
	Reduce both public and private water consumption through the implementation of water conservation practices
	Partner with adjoining local governments to protect City water supply through the implementation of well field protection plans
	Develop a new Waterfront Master Plan that strongly considers public access to the waterfront
Ecological Footprint/Green Space	Reduce commuter traffic by developing an integrated transportation plan that considers the relationship between different modes of transportation (trails, bike paths, public transit <i>etc.</i>)
	Develop planning policies to protect the connectivity of green spaces and existing vegetation
	Develop a tree-protection policy for existing parks and street trees
Green House Gas	Continue with Partners for Climate Change Protection program and set GHG emission reduction targets
	Encourage the use of alternative energy technology and design in land and building development
	Conduct energy audits of municipal buildings/infrastructure to identify

	avenues to improve energy efficiency
Food Security	Develop a public education campaign about the overall implications of sustainable foods and food security
	Work in partnership with community based organizations to develop a community garden
	Develop a procurement policy or buy local food mandate for the City

Table 8: Environmental Pillar. (Source: ICSP, City of Charlottetown, 2010)

Pillar 2: Economic Consideration

	Issues	Actions
Creative Business	Promotethedowntownthroughinfrastructurethatpromotessocialculture (street music, art installations, and art)	
	Partner with community-based business development organizations to focus on business skills development and training for artists, young people and newcomers	
	Develop a common area which enables artists to come together to present and sell their art in a central location	
	Allow new entrepreneurs and artists to use available kiosk units to help reduce their operating costs	
Infrastructure	Develop a strategic replacement plan for aging water, sewage and road infrastructure	
	Develop a City-wide storm water management plan and corresponding site design storm water management policies	
	Review a City policy and planning framework to ensure that City infrastructure and development planning is more pedestrian and active transportation centered	
Community Partnership	Establish a working group, of business, academic, non-profit and public stakeholders, to create a common vision from divergent mandates	
	Partner with stakeholder groups to share and promote sustainable initiatives underway in the City	

Sustainable Development	Undertake a review of the City Official Plan, Zoning Bylaw, Subdivision Bylaw and Engineering Standards to ensure ideas contained in the ICSP are reflected in these documents
	Review the Official Plan actions with responsible parties to ensure that more dense development is permitted in the core so as to discourage urban sprawl
	Meet with developers and the community to discuss and remedy any potential barriers which would discourage the reuse and reclamation of existing buildings and/or brownfield sites

Table 9: Economic Pillar. (Source: ICSP, City of Charlottetown, 2010)

Pillar 3: Social Consideration

Issues	Actions
Social Network	Include a directory of events for local established community groups including a listing of current events, meetings <i>etc.</i>
	Focus on establishing/improving social spaces in the City (City amenities)
	Develop new community information packages (transit, parking, access to City Hall, neighborhood gatherings, and social groups)
	Involve citizens in the plan for the 2014 Celebration of Birthplace of Confederation
Community Ownership of the Plan	Develop a participatory monitoring and evaluation plan that includes a strategy for community education and communication
	City sponsorship of public speakers (professionals on Sustainability) to educate public on relevant sustainability issues.
	Develop workshops for all City staff so they have a clear understanding of the ICSP and how it applies to their role
Safety and Security	Review planning policies to ensure that safety considerations are present in the design of all developments.
	Continue to pursue the recommendations contained in the Police Services Strategic Plan 2009-2011.
Services	Develop municipal partnerships and outreach with various social community groups to assess whether accessibility needs for all citizens are being met

	(including assessing housing needs both for physical and financial capacity).
	Develop select multi-lingual publications through partnerships with community groups.
	Undertake review of City policy and programming to ensure needs for aging population is met.

Table 10: Social Pillar. (Source: ICSP, City of Charlottetown, 2010)

Pillar 4: Cultural Consideration

Issues	Actions
Cultural Events in Winter	Partner with local artists and musicians, schools and to deliver community art classes.
	Partner with academic institutions to host a sustainability lecture series.
	Work with artist/cultural groups to develop a gallery or cultural tour.
	Use existing public transportation infrastructure to offer public transportation from City Centre to/from outside City to increase attendance at events.
Preserve Heritage	Involve more groups (<i>i.e.</i> , aboriginal, Acadian, Immigrants) in heritage events.
	Increase knowledge of island heritage by advocating for a provincial museum.
	Promote traditional skills and way of life (knitting, building, food production/preparation, <i>etc.</i>) through community workshops and demonstrations.
Lifestyles	Ensure there are physical activities that respond to the needs of different demographics included in City programming.
	Better promote the City Activity Program Booklet.
	Seek out and promote community active life style groups (<i>i.e.</i> , walking group/hiking groups).
	Groom more cross country skiing and snow shoeing trails in the City.

Table 11: Cultural Pillar. (Source: ICSP, City of Charlottetown, 2010)

Appendix D: Richmond County MCCAP Template

Based on the ICSP work Richmond County have done in 2010, this appendix is prepared as a template for the MCCAP, which is a new version of ICSP with more action planning notions. There are mainly 6 steps in preparing an MCCAP, which are adaptation committee, climate change hazards & issues, affected locations, facilities & infrastructure, four pillars, and priorities for adaptation. Step five is the part that can be adopted from the work they have done in ICSP. Other information to fill out this template is from data on their website and some other research results. The detailed steps of this template are as below.

Step One: Adaptation Committee

Adaptation Committee Structure	
CHAIR: Electing from the Councilors from ten districts	
Vice Chair: Electing from the Councilors from ten districts	
Members at Large: Members are elected from Richmond County's departments and categorized by sustainable pillars.	1. Environmental Pillar Department of Recreation, Leisure & Community Relations
	2. Economic Pillar Department of Finance Department of Tourism & Economic Development
	3. Social Pillar Department of Public Works Department of Recreation, Leisure & Community Relations
	4. Cultural Pillar Department of Tourism & Economic Development

Table 12: Suggested Committee Structure. (Source: Staff Directory. Richmond County's Website)

Step Two: Climate Change Hazards & Issues

Richmond County	Hazards	Issues
	Severe Storm Sea Level Rise Coastal Erosion	Damage to unique transportation, potential isolation caused by infrastructure damage

Table 13: Hazards & Issues. (Source: December 2011, C-Change Presentation)

Step Three: Affected Locations

Locations	Issues
1. Arichat Waterfront	Coastal Erosion Infrastructure: Seafood industry, sewer facilities
2. Petit de Grat Packers	Highly vulnerable Impact to local industry
3. The Little Anse Breakwater	Highly vulnerable Noticeable erosion Isolation due to highways damage Sea level rise
4. Lennox Passage Bridge	Critical access to the Island Isolation of emergency and tourism
5. Creighton Island	Inhabitants relocation

Table 14: Affected Locations in Isle Madame. (Source: 2011-2012, Isle Madame Research Project)

Step Four: Facilities and Infrastructure

Facilities	Infrastructure
Field Hand Maintenance Sidewalks & Snow Removal Building Maintenance Richmond County Arena Heritage Properties	Water System Sewer System Recycling & Composting Solid Waste Management

Table 15: Facilities and Infrastructure. (Source: Public Works, Richmond County's Website)

Step Five (a): Environmental Consideration

Issues	Actions
Sea Level Rise	Improve the Bras d'Or Lake's watershed according to the vulnerable positions and issues.
Coastal and Shoreline Erosion	Coastline protection in planned areas. Review strategies to sea level rise and water quality.

Step Five (b): Economic Consideration

Issues	Actions
Tourism Promotions	Unified county-wide tourism promotion program
	“Welcome to Richmond County” program
	Build visitor information center, office space and hub
Maximize Benefits	Develop packaging opportunities to assess cross-promotion and marketing opportunities
	Coordinate festivals and event timing to avoid duplication and encourage local attendance
	Developing tourism around individual niches, to avoid competition between communities
Tourism Potential	Municipality will partner with Parks Canada to develop land-based opportunities at St. Peter’s Canal site
	Developing and promote additional tourism amenities for Point Michaud Beach
	Developing and promote local “hidden gems” such as Hills Lake Falls
Tourist Accommodation	Exploring opportunities to develop campgrounds
	Supporting the development of a recreational vehicle park and associated amenities in Richmond County
	Remain open later in the Fall to provide accommodations for the increasing autumn tourist trade
Cape Breton Experience	Easy web access to Richmond County destinations in a greater Cape Breton context
	Supporting the Cape Breton’s Heritage Coast concept as a marketing theme
	Holding dinner/dance events
	Providing wireless internet hotspots
Tourist Potential and Visitor Experiences	Giving people some local context and provide a more complete picture of Richmond County’s cultural tapestry
	Customer service training

Residents and Visitors' Interest	Developing landscape guidelines for the treatment of industrial areas
Industrial Projects	Adding parking or interpretive boards
	Promoting inter-municipal co-operation regarding the regionally significant port developments proposed in Guysborough County
	Creating a marketing partnership with Guysborough County
	Generating academic, industrial and commercial activity in the region
Fishing Industry	Installing 3-phase power to assist the Little Harbor Wharf
	Promoting fish markets and wharf side selling operations in Richmond County to tourists
	Ensuring repairs and maintenance to small craft harbors occur as needed
	Improving cooperation and coordination between harbor authorities regarding maintenance, capacity building and diversified uses
Green Industry	Supporting the retrofitting of municipal buildings for energy efficiency and the use of renewable energy technology
	Installing heat recovery equipment at the coal power plant and/or pulp and paper mill
	Wind turbine development for small and large scale turbines
Land Use	Extending the commercial designation up Corbett's Road in St. Peters with the extension of municipal services
Tourists Parking	Ensuring land use planning regulations are not constricting local commercial interests
	Improving parking in St. Peter's village core
Small Scale Businesses	Supporting the expansion of broadband internet services

Marketing Approach New Residents	Supporting for small businesses in areas such as e-commerce, web design, developing business plans and navigating the bureaucracy Creating a regional professional services group to share expertise on development as the need arises Supporting the S-HRDA Clarifying roles and mandates to avoid duplication of services and maximize the efficiency of the services they provide Municipality will work to define residential opportunities within a 10-20 kilometer radius of Point Tupper and investigate how to best accommodate this future potential growth. Municipality will investigate providing passive (trails) and active (new playground) recreation infrastructure in the Evanston area to create community space and encourage people to move there as a result of possible Point Tupper industrial expansion.
Transportation Networks	Ensuring the construction of the Port Hawkesbury Highway 104 By-pass Ensuring that the Port Hawkesbury Airport continues to operate on a year round basis and should explore the possibility of an industrial sector contribution to maintaining its continued winter operation Investigating upgrades to Highway 4 through St. Peters, to improve the general condition of the road and remove dangerous curves to make it safer for pedestrians, cars and the large number of trucks which pass through the village Opportunities and threats of developing Highway 104 By-pass Supporting the Strait Area Transit initiative, recognizing the benefits it provides to residents in cost savings in transportation Supporting the Strait Area Transit initiative, recognizing the benefits it provides to residents in cost savings in transportation The Municipality will work with the Cape Breton Regional Municipality to encourage additional rail traffic in Cape Breton and present the business case for rail on the Island

Land Resources and Development	The Municipality will conduct an internal inventory of municipal lands within Richmond County

Table 17: Economic Pillar. (Source: Richmond, 2010)

Step Five (c): Social Consideration

Physical Activity and Recreation	The Municipality will work with Richmond County's community organizations to develop additional recreational programs in area The Municipality will investigate the development of a Community Gardening Program Municipality will investigate the feasibility of implementing a Seniors racquet sports program The Municipality will schedule concurrent physical activity opportunities to allow families to participate together or at the same time The Municipality will increase the number of certified tumblebugs leaders, trainers and programs in Richmond County and continue to facilitate tumblebug training for parents, early childhood professionals, etc. The Municipality will work with the Strait Regional School Board to investigate holding an annual "Active Healthy Living Fair" for all schools in Richmond County The Municipality will work with the Board of the Richmond Arena to reconfigure the facility as a Community Centre with a mandate beyond the ice arena The Municipality will work with the Board of the Richmond Arena to broaden the Arena's winter offerings to appeal to different interest and age groups The Municipality will work with the Board of the Richmond Arena to investigate expanding delivery of programs at the Arena to be all-season and more inclusive of the changing demographics of users and potential users The Municipality will identify and promote physical activity opportunities that do not require registration, advanced skills or other commitments Concurrent to the above strategy, the Municipality
----------------------------------	--

	will investigate hiring adult activity supervisors or recruit volunteer activity supervisors to oversee “drop in” activities The Municipality will work with local businesses to encourage the installation of bike racks or other infrastructure at these places of work to encourage physical activity The Municipality will review and where possible revise their current policies to support physical activity initiatives The Municipality will work with the St. Peters Economic Development Organization to expand the St. Peters trail network The Municipality will promote walking and running clubs in Richmond County Schools, linked to Walkabout and Doctors Nova Scotia The Municipality will investigate the development of a Nature Education Program which can be linked to Provincial and local trails The Municipality will work to develop an updated Richmond County Outdoor Recreation Map to encourage self guided outdoor pursuits for residents and tourists The Municipality will work with local economic development agencies and community groups to encourage and support waterfront activities The Municipality will support land purchases or policies that promote coastal access through trails or other uses
Community Well-being	In the event that a new Visitor Information Centre is built in St. Peters, the Municipality will investigate the feasibility of incorporating a new public library for the community into the facility The Municipality will work with local health care providers and other levels of government to develop a strategy to recruit and retain physicians and other health care professionals in Richmond County Given its importance to the region and to ensure its continued operation, the Municipality will assist the Strait Richmond Hospital’s fundraising when necessary The Municipality will investigate a partnership with the Strait Regional School Board and other agencies to provide and promote expanded and accessible

	adult education and literacy programs in Richmond County The Municipality will initiate and develop a strategy to identify vulnerable groups such as seniors, persons with disabilities and the economically challenged and seek ways or programs to allow their continued participation in their community The Municipality will continue to support Strait Area Transit and encourage its expansion to underserved areas such as Chapel Island and Samsonville The Municipality will investigate the creation of a centralized carpool registration system to allow interested participants to contact one another The Municipality will investigate the feasibility of working with and sharing certain services with the Chapel Island First nation for the benefit of all residents
Housing	The Municipality will continue to require that all planning documents in Richmond County contain provisions for affordable housing, special needs housing, manufactured housing and rental accommodation The Municipality will initiate a housing options study to determine if there is a need for diversified housing options, including affordable and temporary housing and how to address this need The Municipality will partner with other government and non-government agencies to provide assisted housing in Richmond County where the demand is highest
Community Safety	The Municipality will support upgrades to Highway 4 through the Village of St. Peter's, to improve the general condition of the road and remove dangerous curves to make it safer for pedestrians, cars and the large number of trucks which pass through the village The Municipality will continue to work with the Royal Canadian Mounted Police to ensure the continued police protection for residents of the County The Municipality will continue to work with and support the operation of the various volunteer fire departments in Richmond County

	The Municipality will work with and lobby the Emergency Management Office of Nova Scotia that to ensure that emergency communications are available in all parts of Richmond County regardless of whether there is a power outage
--	--

Table 18: Social Pillar. (Source: Richmond, 2010)

Step Five (d): Cultural Consideration

Preserving Richmond's Built Heritage	The Municipality will conduct an inventory of heritage properties in Richmond County to better plan their protection
Preserving Richmond's Cultural Heritage	The Municipality will conduct an inventory of museums in Richmond County. Within this inventory, agencies that do not have web capabilities shall be identified with the aim of linking them to agencies in the Province who can help them build, upgrade and maintain their websites The Municipality will encourage museums to look at resource sharing to allow the hiring of permanent staff so as not to need to train new staff every year The Municipality will continue to work with the local artistic community to ensure that resources and facilities are available to them to support their ongoing work The Municipality will work with cultural groups and other levels of government to provide widely accessible opportunities for residents to learn English, French, Gaelic, Mi'kmaq and other languages spoken in Richmond The Municipality will co-ordinate opportunities for Richmond County's cultural organizations to meet each other for the purposes of learning, planning and product development
Population Retention	The Municipality will work with local economic development agencies to develop a website for the Richmond County as a whole or for specific communities to promote the area as a place to live and succeed The Municipality will work with local economic development agencies to target and attract youth to the community The Municipality will work with local economic

	development agencies to initiate a welcome wagon/community information package for new and prospective residents The Municipality will continue to work with schools, social agencies and economic development agencies with the intention of promoting Richmond County as an attractive place to live at the conclusion of high school
--	---

Table 19: Cultural Pillar. (Source: Richmond, 2010)

Step Six: Priorities for Adaptation

Priority Rating	Action/Strategy	Sustainability Benefit	Total Cost
1	Pre-design work for construction of water treatment plant for Louisdale/Evanston/Whiteside (addressing elevated manganese and iron in water)	Cleaner water for communities	\$25,000
2	Upgrade or replace pumping stations in Arichat system to increase capacity and reduce sewage overflow	Reduced sewage outflow into water bodies	\$750,000
3	Pre-design work for District 4 Wastewater Collection and Treatment System	Cleaner water, reduced reliance on septic	\$35,000
4	Replace inoperable Evanston Wastewater Treatment Plant	Protects watercourses by ensuring wastewater is treated	\$2,250,000
5	Comprehensive leak detection and repair program for Richmond Water Utility	Enhances efficiency of water system	\$150,000
6	Manhole inspection and rehabilitation program to reduce level of extraneous water entering system	Ensures cleaner environment by ensuring efficiency of system	\$250,000
7	Inflow/infiltration study in Petit de Grat system to assess condition of piping, manholes, pumping stations	Ensures cleaner environment by ensuring efficiency of system	\$90,000

8	Implement key recommendations in Source Water Protection Plan for Richmond County, including land acquisition and re-designating Water Supply areas	Ensures cleaner water for residents by protecting watershed areas	\$175,000
9	Risk assessment studies for each wastewater treatment plant to ensure compliance with future Canadian Council of Ministers of the Environment Strategy for Municipal Wastewater Effluent	Ensures cleaner environment by ensuring efficiency of system	\$200,000
10	Wastewater Control Study for District 3	Cleaner water, reduced reliance on septic	\$40,000
11	Install attractive and coordinated way finding signage throughout County to market attractions and amenities	Encourage tourism, provides increased knowledge of events/venues	\$100,000
12	Construct 6,600 sq. Visitor information center in St. Peters to include interpretive center and hub for Cape Bretons Heritage Coast	Encourage tourism, provides increased knowledge of events/venues	\$3,125,000

Table 20: Project Priorities. (Source: Richmond, 2010)

Appendix E: Community Adaptation Mobile Application Code

This appendix lists all the code used to develop a community adaptation mobile application. The programming language used in this development is called Objective-C, which is an object-oriented programming language specifically used for Apple iOS device application development, like iPhone, iPad, and iPod Touch. The concept of iOS application development is based on the MVC (Model, View, Controller). Model means the application itself, view is how the application presents to users, and controller is how application response to users request. All the code listed in the following, is from different controllers in the application, which implement different functions. Based on the controller's name, the code is listed as follows,

HomeView

```
#import <UIKit/UIKit.h>

@interface HomeViewController : UIViewController{
}

@property (strong, nonatomic) IBOutlet UILabel *notificationlabel;

@end

#import "HomeViewController.h"
#import <Parse/Parse.h>
#import <QuartzCore/QuartzCore.h>
#import "CLTickerView.h"

static NSString *areanotifi;
static NSString *placenotifi;

@interface HomeViewController ()

@end

@implementation HomeViewController
@synthesize notificationlabel;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
    }
    return self;
}
```

```

- (void)viewDidLoad
{
    [super viewDidLoad];

}

-(void)viewWillAppear:(BOOL)animated {

    [self.notificationlabel.subviews makeObjectsPerformSelector:
    @selector(removeFromSuperview)];

    PFQuery *query1 = [PFQuery queryWithClassName:@"notification"];

    [query1 findObjectsInBackgroundWithBlock:^(NSArray *objects, NSError *error) {
        if (!error) {

            NSString *total=@"NOTIFICATION: ";

            for (int i=0; i<objects.count; i++) {

                areanotifi=[ [objects objectAtIndex:i] objectForKey:@"area"];
                placenotifi=[ [objects objectAtIndex:i]
                objectForKey:@"place"]stringByAppendingString: @" #";
                total=[total stringByAppendingString:[ areanotifi
                stringByAppendingString:placenotifi  ]];

            }

            CLTickerView *ticker = [[CLTickerView alloc] initWithFrame:CGRectMake(0, 20,
            320, 30)];
            ticker.marqueeFont = [UIFont boldSystemFontOfSize:16];
            ticker.marqueeStr =total;
            [self.notificationlabel addSubview:ticker];

        } else {

            NSLog(@"Error: %@ %@", error, [error userInfo]);

        }

    }];

}

-(void)viewDidAppear:(BOOL)animated{

}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];

}

@end

```

MapView


```

#import <UIKit/UIKit.h>
#import<MapKit/MapKit.h>

@interface MapViewController : UIViewController<MKMapViewDelegate>
@property (strong, nonatomic) IBOutlet MKMapView *mapview;

@property (weak, nonatomic) IBOutlet UIButton *vulnerableplaces;
@property (weak, nonatomic) IBOutlet UIButton *dangersareas;

@property (weak, nonatomic) IBOutlet UISegmentedControl *fourpillars;
@property (weak, nonatomic) IBOutlet UISegmentedControl *fiveplaces;
@property (weak, nonatomic) IBOutlet UISegmentedControl *areasandplaces;
@property (weak, nonatomic) IBOutlet UISegmentedControl *helpandhelper;
@property (weak, nonatomic) IBOutlet UISegmentedControl *maptype;
@property (weak, nonatomic) IBOutlet UILabel *areainfor;

@end

#import "MapViewController.h"
#import<MapKit/MapKit.h>
#import <Parse/Parse.h>
#import <UIKit/UIKit.h>
#import <QuartzCore/QuartzCore.h>
#import "MyAnnotation.h"

//static BOOL haveAlreadyReceivedCoordinates = NO;

NSString *pillar;
NSString *place;
NSString *envir;
NSString *eco;
NSString *soc;
NSString *cul;
static int tag;

int apoint;
static int iconnumber;
NSString *nameoficon[0];
CLLocationCoordinate2D coordss[0];

@interface MapViewController ()

@end

@implementation MapViewController
@synthesize mapview=_mapview;
@synthesize vulnerableplaces;
@synthesize dangersareas;
@synthesize fourpillars;
@synthesize fiveplaces;
@synthesize areasandplaces;
@synthesize helpandhelper;
@synthesize maptype;
@synthesize areainfor=_areainfor;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {

    }
    return self;
}

```

```

}

- (void)viewDidLoad
{
    [super viewDidLoad];
    _mapview.showsUserLocation = YES;
    _mapview.zoomEnabled = YES;
    _mapview.scrollEnabled = YES;
    _mapview.mapType = MKMapTypeStandard;
    _mapview.delegate=self;
    fourpillars.hidden=YES;
    fiveplaces.hidden=YES;
    areasandplaces.hidden=YES;
    helpandhelper.hidden=YES;

}

-(void)viewDidAppear:(BOOL)animated
{

    CLLocationCoordinate2D p;
    p.latitude=45.550815;
    p.longitude=-61.023674;
    MKCoordinateRegion mapRegion;
    mapRegion.center = p;
    mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    [_mapview setRegion:mapRegion animated: YES];

}

- (IBAction)currentlocation:(UIButton *)sender {

    [_mapview setCenterCoordinate:_mapview.userLocation.location.coordinate
    animated:YES];
}

- (IBAction)annotation:(UIButton *)sender {
    if (fourpillars.hidden==YES) {
        fourpillars.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [fourpillars.layer addAnimation:animation forKey:nil];
        fourpillars.hidden=NO;
        [fourpillars setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {

        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [fourpillars.layer addAnimation:animation forKey:nil];
        fourpillars.hidden=YES;
        [_mapview removeAnnotations:[_mapview annotations]];
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
    }
}

```

```

        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
        [_mapview setRegion:mapRegion animated: YES];
        fourpillars.hidden=YES;
    }
}

- (IBAction)vulnerableplaces:(UIButton *)sender {
    if (fiveplaces.hidden==YES) {
        fiveplaces.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [fiveplaces.layer addAnimation:animation forKey:nil];
        fiveplaces.hidden=NO;
        fiveplaces.selected=NO;
        [fiveplaces setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [fiveplaces.layer addAnimation:animation forKey:nil];
        fiveplaces.hidden=YES;
        [_mapview removeOverlays:[_mapview overlays]];
        self.areasinfor.text=nil;
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
        [_mapview setRegion:mapRegion animated: YES];
        fiveplaces.hidden=YES;
    }
}

- (IBAction)dangersareas:(UIButton *)sender {
    if (areasandplaces.hidden==YES) {
        areasandplaces.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [areasandplaces.layer addAnimation:animation forKey:nil];
        areasandplaces.hidden=NO;
        areasandplaces.selected=NO;
        [areasandplaces setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [areasandplaces.layer addAnimation:animation forKey:nil];
        areasandplaces.hidden=YES;
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    }
}

```

```

        [_mapview setRegion:mapRegion animated: YES];
        [_mapview removeOverlays:[_mapview overlays]];
        [_mapview removeAnnotations:[_mapview annotations]];

        areasandplaces.hidden=YES;
    }
}

- (IBAction)usershare:(UIButton *)sender {
    if (helpandhelper.hidden==YES) {
        helpandhelper.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [helpandhelper.layer addAnimation:animation forKey:nil];
        helpandhelper.hidden=NO;
        helpandhelper.selected=NO;
        [helpandhelper setSelectedSegmentIndex:UISegmentedControlNoSegment];
    }else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [helpandhelper.layer addAnimation:animation forKey:nil];
        helpandhelper.hidden=YES;
    }
}

- (IBAction)fourpillars:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;
    switch ([segmentedControl selectedSegmentIndex]) {
        case 0:
        {
            tag=1;

            [_mapview removeAnnotations:[_mapview annotations]];

            PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
            [query0 whereKey:@"pillar" equalTo:@"environmental"];
            [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError
*error) {
                if (!error) {
                    CLLocationCoordinate2D annoo;
                    iconnumber=objectss.count;
                    for (int i=0; i<objectss.count; i++)

                    { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                        annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                        annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                        aaa.coordinate=annoo;
                        aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;
                        aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];

```

```

        aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

        [_mapview addAnnotation:aaa];
    }

    } else {
        // Log details of the failure
        NSLog(@"Error: %@ %@", error, [error userInfo]);
    }

    }];

}

break;
case 1:
{
    tag=2;

    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"economic"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

            { MyAnnotation *aaa=[[MyAnnotation alloc] init];

                annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                aaa.coordinate=annoo;
                aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;
                aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                [_mapview addAnnotation:aaa];

            }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];
}

```

```

    }
    break;

case 2:
{
    tag=3;
    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"social"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                    aaa.coordinate=annoo;
                    aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;
                    aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                    aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                    [_mapview addAnnotation:aaa];

                }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

break;

case 3:
{
    tag=4;
    [_mapview removeAnnotations:[_mapview annotations]];
    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"cultural"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

```

```

        annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
        annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

        aaa.coordinate=annoo;
        aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;
        aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
        aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

        [_mapview addAnnotation:aaa];

    }

    } else {
        // Log details of the failure
        NSLog(@"Error: %@ %@", error, [error userInfo]);
    }

    }];

}

break;

case 4:
{
    tag=0;
    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query1 = [PFQuery queryWithClassName:@"EditMap"];

    [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)
            {
                MyAnnotation *aaa=[[MyAnnotation alloc] init];

                annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

                aaa.coordinate=annoo;
                aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;
                aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                [_mapview addAnnotation:aaa];

            }

```

```

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

break;

case UISegmentedControlNoSegment:
    // do something

    break;
default:
    NSLog(@"No option for: %d", [segmentedControl selectedSegmentIndex]);
}

}

- (IBAction)fiveplaces:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;
    switch ([segmentedControl selectedSegmentIndex]) {
        case 0:
            { [_mapview removeOverlays:[_mapview overlays]];

                CLLocationCoordinate2D coords[9]={
                    CLLocationCoordinate2DMake(45.507883, -61.042528),
                    CLLocationCoordinate2DMake(45.510199, -61.041669),
                    CLLocationCoordinate2DMake(45.510981, -61.033859),
                    CLLocationCoordinate2DMake(45.513206, -61.028537),
                    CLLocationCoordinate2DMake(45.516394, -60.994677),
                    CLLocationCoordinate2DMake(45.512635, -60.990471),
                    CLLocationCoordinate2DMake(45.510379, -60.998368),
                    CLLocationCoordinate2DMake(45.511733, -61.001758),
                    CLLocationCoordinate2DMake(45.507883, -61.042528),
                };

                MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:9];
                [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.512996,-
61.014161)];
                [_mapview addOverlay:polyLine];
                MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
                [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.512996,-61.014161), span)
animated:YES];
                self.areaInfor.text=@"Arichat Waterfront:High volume of historical photos
available due to its importance in the past Populated area on Isle Madame Highly utilized
area Area of noticeable erosion Important infrastructure: Substantial seafood industry
holdings in this area (Government wharf, Premium Seafood Inc. wharf and infrastructure),
town sewer facilities.";
            }

            break;
        case 1:
            {
                [_mapview removeOverlays:[_mapview overlays]];

                CLLocationCoordinate2D coords[10]={

```



```

        CLLocationCoordinate2DMake(45.514048, -60.956010),
        CLLocationCoordinate2DMake(45.512725, -60.961461),
        CLLocationCoordinate2DMake(45.510259, -60.964980),
        CLLocationCoordinate2DMake(45.503492, -60.968156),
        CLLocationCoordinate2DMake(45.498950, -60.969915),
        CLLocationCoordinate2DMake(45.494047, -60.959401),
        CLLocationCoordinate2DMake(45.498860, -60.954508),
        CLLocationCoordinate2DMake(45.504635, -60.956053),
        CLLocationCoordinate2DMake(45.512214, -60.952148),
        CLLocationCoordinate2DMake(45.514048, -60.956010),

    };

    MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:10];
    [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.505898,-
60.960130)];
    [_mapview addOverlay:polyLine];
    MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
    [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.505898,-60.960130), span)
animated:YES];
    self.areasinfor.text=@"Petit de Grat Packers:Area of noticeable impact- Highly
vulnerable Economic impact to local industry: Important seasonal source of income for
many community members (Approximately 90 employees) as well as selling opportunities for
crab fishermen.";

    }
    break;
case 2:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[11]={
        CLLocationCoordinate2DMake(45.506319, -60.944509),
        CLLocationCoordinate2DMake(45.502710, -60.940990),
        CLLocationCoordinate2DMake(45.500845, -60.936355),
        CLLocationCoordinate2DMake(45.496032, -60.935497),
        CLLocationCoordinate2DMake(45.494588, -60.938158),
        CLLocationCoordinate2DMake(45.492693, -60.939531),
        CLLocationCoordinate2DMake(45.491038, -60.933437),
        CLLocationCoordinate2DMake(45.493264, -60.929660),
        CLLocationCoordinate2DMake(45.500664, -60.932750),
        CLLocationCoordinate2DMake(45.503642, -60.933523),
        CLLocationCoordinate2DMake(45.506319, -60.944509),

    };

    MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:11];
    [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.498288,-
60.933995)];
    [_mapview addOverlay:polyLine];
    MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
    [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.498288,-60.933995), span)
animated:YES];
    self.areasinfor.text=@"The Little Anse Breakwater:Highly vulnerable area
Noticeable erosion Residents have experienced isolation through impassable highways
Houses are threatened by sea level rise.";

    }
    break;
case 3:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[7]={
        CLLocationCoordinate2DMake(45.607165, -61.088147),
        CLLocationCoordinate2DMake(45.586325, -61.094584),
        CLLocationCoordinate2DMake(45.576113, -61.069350),

```

```

        CLLocationCoordinate2DMake(45.582181, -61.038708),
        CLLocationCoordinate2DMake(45.601580, -61.044545),
        CLLocationCoordinate2DMake(45.608126, -61.059394),
        CLLocationCoordinate2DMake(45.607165, -61.088147),

};

MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:7];
[_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.593773,-
61.067290)];
[_mapview addOverlay:polyLine];
MKCoordinateSpan span = MKCoordinateSpanMake(0.1, 0.1);
[_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.593773,-61.067290), span)
animated:YES];
        self.areasinfor.text=@"Lennox Passage Bridge:Critical access to the island:
Only entering and exiting point for the entire island for inhabitants. Emergency access,
tourism access etc..";
    }
    break;
case 4:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[11]={
        CLLocationCoordinate2DMake(45.514680, -61.091837),
        CLLocationCoordinate2DMake(45.517747, -61.090164),
        CLLocationCoordinate2DMake(45.516935, -61.084199),
        CLLocationCoordinate2DMake(45.520243, -61.082911),
        CLLocationCoordinate2DMake(45.511101, -61.069178),
        CLLocationCoordinate2DMake(45.509657, -61.070037),
        CLLocationCoordinate2DMake(45.511131, -61.076088),
        CLLocationCoordinate2DMake(45.513507, -61.080207),
        CLLocationCoordinate2DMake(45.514650, -61.083083),
        CLLocationCoordinate2DMake(45.513056, -61.085186),
        CLLocationCoordinate2DMake(45.514680, -61.091837),

    };

    MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:11];
    [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.515552,-
61.079950)];
    [_mapview addOverlay:polyLine];
    MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
    [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.515552,-61.079950), span)
animated:YES];
        self.areasinfor.text=@"Creighton Island:Area of noticeable impact: The area
has been cut off forcing inhabitants to relocate off the island within a small amount of
time.";
    }
    break;

case 5:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords1[9]={
        CLLocationCoordinate2DMake(45.507883, -61.042528),
        CLLocationCoordinate2DMake(45.510199, -61.041669),
        CLLocationCoordinate2DMake(45.510981, -61.033859),
        CLLocationCoordinate2DMake(45.513206, -61.028537),
        CLLocationCoordinate2DMake(45.516394, -60.994677),
        CLLocationCoordinate2DMake(45.512635, -60.990471),
        CLLocationCoordinate2DMake(45.510379, -60.998368),

```

```

        CLLocationCoordinate2DMake(45.511733, -61.001758),
        CLLocationCoordinate2DMake(45.507883, -61.042528),
    };
    MKPolyline *polyLine1=[MKPolyline polylineWithCoordinates:coords1 count:9];
    [_mapview addOverlay:polyLine1];

    CLLocationCoordinate2D coords2[10]={
        CLLocationCoordinate2DMake(45.514048, -60.956010),
        CLLocationCoordinate2DMake(45.512725, -60.961461),
        CLLocationCoordinate2DMake(45.510259, -60.964980),
        CLLocationCoordinate2DMake(45.503492, -60.968156),
        CLLocationCoordinate2DMake(45.498950, -60.969915),
        CLLocationCoordinate2DMake(45.494047, -60.959401),
        CLLocationCoordinate2DMake(45.498860, -60.954508),
        CLLocationCoordinate2DMake(45.504635, -60.956053),
        CLLocationCoordinate2DMake(45.512214, -60.952148),
        CLLocationCoordinate2DMake(45.514048, -60.956010),
    };

    MKPolyline *polyLine2=[MKPolyline polylineWithCoordinates:coords2 count:10];
    [_mapview addOverlay:polyLine2];

    CLLocationCoordinate2D coords3[11]={
        CLLocationCoordinate2DMake(45.506319, -60.944509),
        CLLocationCoordinate2DMake(45.502710, -60.940990),
        CLLocationCoordinate2DMake(45.500845, -60.936355),
        CLLocationCoordinate2DMake(45.496032, -60.935497),
        CLLocationCoordinate2DMake(45.494588, -60.938158),
        CLLocationCoordinate2DMake(45.492693, -60.939531),
        CLLocationCoordinate2DMake(45.491038, -60.933437),
        CLLocationCoordinate2DMake(45.493264, -60.929660),
        CLLocationCoordinate2DMake(45.500664, -60.932750),
        CLLocationCoordinate2DMake(45.503642, -60.933523),
        CLLocationCoordinate2DMake(45.506319, -60.944509),
    };

    MKPolyline *polyLine3=[MKPolyline polylineWithCoordinates:coords3 count:11];
    [_mapview addOverlay:polyLine3];

    CLLocationCoordinate2D coords4[7]={
        CLLocationCoordinate2DMake(45.607165, -61.088147),
        CLLocationCoordinate2DMake(45.586325, -61.094584),
        CLLocationCoordinate2DMake(45.576113, -61.069350),
        CLLocationCoordinate2DMake(45.582181, -61.038708),
        CLLocationCoordinate2DMake(45.601580, -61.044545),
        CLLocationCoordinate2DMake(45.608126, -61.059394),
        CLLocationCoordinate2DMake(45.607165, -61.088147),
    };

    MKPolyline *polyLine4=[MKPolyline polylineWithCoordinates:coords4 count:7];
    [_mapview addOverlay:polyLine4];

    CLLocationCoordinate2D coords5[11]={
        CLLocationCoordinate2DMake(45.514680, -61.091837),
        CLLocationCoordinate2DMake(45.517747, -61.090164),
        CLLocationCoordinate2DMake(45.516935, -61.084199),
        CLLocationCoordinate2DMake(45.520243, -61.082911),
        CLLocationCoordinate2DMake(45.511101, -61.069178),
        CLLocationCoordinate2DMake(45.509657, -61.070037),
        CLLocationCoordinate2DMake(45.511131, -61.076088),
        CLLocationCoordinate2DMake(45.513507, -61.080207),
        CLLocationCoordinate2DMake(45.514650, -61.083083),
        CLLocationCoordinate2DMake(45.513056, -61.085186),
    };

```

```

        CLLocationCoordinate2DMake(45.514680, -61.091837),

};

MKPolyline *polyLine5=[MKPolyline polylineWithCoordinates:coords5 count:11];
[_mapview addOverlay:polyLine5];
self.areasinfor.text=nil;

MKCoordinateSpan span = MKCoordinateSpanMake(0.18, 0.18);
[_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.550815,-61.023674), span)
animated:YES];

}

        break;
    case UISegmentedControlNoSegment:
        // do something

        break;
    default:
        NSLog(@"No option for: %d", [segmentedControl selectedSegmentIndex]);
}

- (IBAction)areasandplaces:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;

    switch([segmentedControl selectedSegmentIndex])
    {
        case 0:
        {
            PFQuery *query2 = [PFQuery queryWithClassName:@"Dangersareas"];

            [query2 findObjectsInBackgroundWithBlock:^(NSArray *objects, NSError *error)
            {
                if (!error) {

                    for (int i=0; i<objects.count; i++) {

                        NSArray *ff;
                        ff= [ [objects objectAtIndex:i] objectForKey:@"latitude"];

                        NSArray *hh;
                        hh= [ [objects objectAtIndex:i] objectForKey:@"longtitude"];

                        apoint =[ [objects objectAtIndex:i] objectForKey:@"index"]
integerValue];

                        for (int i=0; i<apoint; i++) {
                            coordss[i].latitude = [[ff objectAtIndex:i] floatValue];
                        }
                        for (int i=0; i<apoint; i++) {
                            coordss[i].longitude=[[hh objectAtIndex:i] floatValue ];
                        }
                        MKPolygon *polygon;
                        polygon = [MKPolygon polygonWithCoordinates:coordss count:apoint];

                        [_mapview addOverlay:polygon];

                        CLLocationCoordinate2D points[2];
                        points[0]= CLLocationCoordinate2DMake(coordss[0].latitude,
coordss[0].longitude);

```

```

        points[1]= CLLocationCoordinate2DMake(coordss[apoint-1].latitude,
coordss[apoint-1].longitude);
        MKPolyline *polyline;
        polyline = [MKPolyline polylineWithCoordinates:points count:2];
        [_mapview addOverlay:polyline];
    }
    } else {
        // Log details of the failure
        NSLog(@"Error: %@ %@", error, [error userInfo]);
    }
}];

}

break;
case 1:
{
    tag=5;
    PFQuery *query2 = [PFQuery queryWithClassName:@"Dangersplaces"];

    [query2 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {

            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *anot=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

                    anot.coordinate=annoo;
                    anot.title=[ [objectss objectAtIndex:i] objectForKey:@"title"];
                    anot.subtitle=[ [objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                    anot.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;

                    [_mapview addAnnotation:anot];

                }

            } else {
                // Log details of the failure
                NSLog(@"Error: %@ %@", error, [error userInfo]);
            }
        }];

    }
    break;
}

```

```

        default:
            break;
    }

}

- (IBAction)helpandhelper:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;

    switch([segmentedControl selectedSegmentIndex]){

        case 0:{
            tag=6;
            [_mapview removeAnnotations:[_mapview annotations]];

            PFQuery *query1 = [PFQuery queryWithClassName:@"share"];

            [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
            {
                if (!error) {
                    CLLocationCoordinate2D annoo;
                    iconnumber=objectss.count;
                    for (int i=0; i<objectss.count; i++)

                        { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                            annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                            annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                            aaa.coordinate=annoo;
                            aaa.iconname=[ [objectss objectAtIndex:i] objectForKey:@"icon"] ;
                            aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                            aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                            [_mapview addAnnotation:aaa];

                        }

                } else {
                    // Log details of the failure
                    NSLog(@"Error: %@ %@", error, [error userInfo]);
                }

            }
        };

    }

    break;
    case 1:{

    }

    break;

    default:
        break;

    }
}

```

```

}

- (IBAction)maptype:(UISegmentedControl *)sender {
    if (maptype.selectedSegmentIndex==0) {
        _mapview.mapType=MKMapTypeStandard;

    }else if (maptype.selectedSegmentIndex==1) {
        _mapview.mapType=MKMapTypeHybrid;
    }else if (maptype.selectedSegmentIndex==2) {
        _mapview.mapType=MKMapTypeSatellite;
    }
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

- (MKOverlayView *)mapView:(MKMapView *)mapView viewForOverlay:(id )overlay
{
    /*MKPolygonView *polygonView ;
    polygonView.lineWidth = 0.5;
    polygonView.strokeColor = [UIColor redColor];
    polygonView.fillColor = [[UIColor colorWithRed:0.9176 green:0.9098 blue:0.8117
alpha:1.0] colorWithAlphaComponent:0.5];

    return polygonView;*/

    MKPolylineView *polyLineView = [[MKPolylineView alloc] initWithOverlay: overlay];
    polyLineView.strokeColor = [UIColor blueColor];
    polyLineView.lineWidth = 4.0;

    //polyLineView.backgroundColor= [UIColor blueColor];
    //polyLineView.alpha=0.3f;

    return polyLineView;
}

- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
<MKAnnotation>)Annotation
{
    MyAnnotation *annoo=(MyAnnotation *)Annotation;

    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    if(Annotation != _mapview.userLocation)
    {
        for (int i=0; i<iconnumber; i++) {

            if (tag==0)
            {

                //indexname[i]= [[NSUserDefaults standardUserDefaults]
objectForKey:@"index"] integerValue];

```

```

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==5)
    {

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

    else if (tag==1)
    {

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==2)
    {

```



```

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==3)
    {

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==4)
    {

        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage * img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==6)
    {

        NSString * name=annoo.iconname;
        NSString *png=@"png";

```

```

        NSString *all=[name stringByAppendingString:png];
        UIImage *img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

}

}
else
    [_mapview.userLocation setTitle:@"I am here"];
return nil;

}

@end

```

MyAnnotation

```

#import <Foundation/Foundation.h>
#import <MapKit/MapKit.h>
@interface MyAnnotation : NSObject<MKAnnotation>{

    CLLocationCoordinate2D coordinate;
    NSString* title;
    NSString* subtitle;

    NSString *iconname;
    NSString *notification;
}
@property (nonatomic, assign) CLLocationCoordinate2D coordinate;
@property (nonatomic, copy ) NSString* title;
@property (nonatomic, copy) NSString* subtitle;

@property (nonatomic, copy) NSString *iconname;
@property (nonatomic, copy) NSString *notification;
@end

#import "MyAnnotation.h"

@implementation MyAnnotation
@synthesize title;
@synthesize subtitle;
@synthesize coordinate;

```

```

@synthesize iconname;
@synthesize notification;

@end

```

LoginView

```

#import <UIKit/UIKit.h>
#import <Parse/Parse.h>
@interface LoginViewController : UIViewController <PFLoginViewControllerDelegate,
PFSignUpViewControllerDelegate, UITextFieldDelegate>

@property (nonatomic,copy) NSString *usernameinput;
@property (nonatomic,copy) NSString *passwordinput;

@property (strong, nonatomic) IBOutlet UIButton *login;

@end

#import "LoginViewController.h"
#import <Parse/Parse.h>

@interface LoginViewController ()
@property (strong, nonatomic) IBOutlet UITextField *username;
@property (strong, nonatomic) IBOutlet UITextField *password;

@end

@implementation LoginViewController

@synthesize username=username;
@synthesize password=password;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (IBAction)login:(UIButton *)sender {

    [PFUser loginWithUsernameInBackground:self.username.text password:self.password.text
    block:^(PFUser *user, NSError *error) {
        if (user) {
            [self performSegueWithIdentifier:@"admin"
sender:self];
        } else {

            UIAlertView *message = [[UIAlertView alloc]
initWithTitle:@"Alert"
message:@"Wrong username or password"
delegate:nil
cancelButtonTitle:@"OK"

```

```

otherButtonTitles:nil];
                                }
                                }];

}

- (void)viewDidLoad
{
    [super viewDidLoad];
    self.username.delegate=self;
    self.password.delegate=self;
}

-(void)viewWillAppear:(BOOL)animated{

    [super viewWillAppear:animated];
    [self.username becomeFirstResponder];
}

-(BOOL)textFieldShouldReturn:(UITextField *)textField{

    return YES;
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end

```

AdminView

```

#import <UIKit/UIKit.h>
#import<MapKit/MapKit.h>
#import <Parse/Parse.h>

@interface AdminViewController : UIViewController<MKMapViewDelegate>
@property (strong, nonatomic) IBOutlet MKMapView *mapview;
@property (strong, nonatomic) IBOutlet UILabel *numberofusers;
@property (strong, nonatomic) IBOutlet UISegmentedControl *fourpillars;
@property (strong, nonatomic) IBOutlet UISegmentedControl *fiveplaces;
@property (strong, nonatomic) IBOutlet UISegmentedControl *areasandplaces;
@property (strong, nonatomic) IBOutlet UISegmentedControl *helpandhelper;
@property (strong, nonatomic) IBOutlet UILabel *areainfor;
@property (strong, nonatomic) IBOutlet UISegmentedControl *maptype;

@end

#import "AdminViewController.h"
#import<MapKit/MapKit.h>
#import <Parse/Parse.h>
#import <UIKit/UIKit.h>
#import <QuartzCore/QuartzCore.h>
#import "MyAnnotation.h"

static int tag;

int apoint;

```

```

static int iconnumber;
NSString *nameoficon[0];
CLLocationCoordinate2D coordss[0];
static BOOL haveAlreadyReceivedCoordinates = NO;

@interface AdminViewController ()

@end

@implementation AdminViewController
@synthesize mapview=_mapview;
@synthesize numberofusers=_numberofusers;
@synthesize fourpillars;
@synthesize fiveplaces;
@synthesize areasandplaces;
@synthesize helpandhelper;
@synthesize areaainfor;
@synthesize maptype;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (IBAction)maptype:(UISegmentedControl *)sender {

    if (maptype.selectedSegmentIndex==0) {
        _mapview.mapType=MKMapTypeStandard;

    }else if (maptype.selectedSegmentIndex==1) {
        _mapview.mapType=MKMapTypeHybrid;
    }else if (maptype.selectedSegmentIndex==2) {
        _mapview.mapType=MKMapTypeSatellite;
    }

}

- (IBAction)annotation:(UIButton *)sender {
    if (fourpillars.hidden==YES) {
        fourpillars.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [fourpillars.layer addAnimation:animation forKey:nil];
        fourpillars.hidden=NO;
        [fourpillars setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {

        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [fourpillars.layer addAnimation:animation forKey:nil];
        [_mapview removeAnnotations:[_mapview annotations]];
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    }
}

```

```

        [_mapview setRegion:mapRegion animated: YES];
        fourpillars.hidden=YES;
    }
}
- (IBAction)vulnerableplaces:(UIButton *)sender {
    if (fiveplaces.hidden==YES) {
        fiveplaces.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [fiveplaces.layer addAnimation:animation forKey:nil];
        fiveplaces.hidden=NO;
        fiveplaces.selected=NO;
        [fiveplaces setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [fiveplaces.layer addAnimation:animation forKey:nil];
        [_mapview removeOverlays:[_mapview overlays]];
        self.areasinfor.text=nil;
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
        [_mapview setRegion:mapRegion animated: YES];
        fiveplaces.hidden=YES;
    }
}
- (IBAction)dangersareas:(UIButton *)sender {
    if (areasandplaces.hidden==YES) {
        areasandplaces.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [areasandplaces.layer addAnimation:animation forKey:nil];
        areasandplaces.hidden=NO;
        areasandplaces.selected=NO;
        [areasandplaces setSelectedSegmentIndex:UISegmentedControlNoSegment];

    }else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [areasandplaces.layer addAnimation:animation forKey:nil];
        CLLocationCoordinate2D p;
        p.latitude=45.550815;
        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
        [_mapview setRegion:mapRegion animated: YES];
        [_mapview removeOverlays:[_mapview overlays]];
        [_mapview removeAnnotations:[_mapview annotations]];
        areasandplaces.hidden=YES;
    }
}
}

```

```

- (IBAction)usershare:(UIButton *)sender {
    if (helpandhelper.hidden==YES) {
        helpandhelper.hidden=NO;
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionMoveIn;
        animation.duration = 0.3;
        [helpandhelper.layer addAnimation:animation forKey:nil];
        helpandhelper.hidden=NO;
        helpandhelper.selected=NO;
        [helpandhelper setSelectedSegmentIndex:UISegmentedControlNoSegment];
    }
    else
    {
        CATransition *animation = [CATransition animation];
        animation.type = kCATransitionFade;
        animation.duration = 0.3;
        [helpandhelper.layer addAnimation:animation forKey:nil];
        helpandhelper.hidden=YES;
    }
}

- (IBAction)fourpillars:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;
    switch ([segmentedControl selectedSegmentIndex]) {
        case 0:
        {
            tag=1;

            [_mapview removeAnnotations:[_mapview annotations]];

            PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
            [query0 whereKey:@"pillar" equalTo:@"environmental"];
            [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
            {
                if (!error) {
                    CLLocationCoordinate2D annoo;
                    iconnumber=objectss.count;
                    for (int i=0; i<objectss.count; i++)

                        { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                            annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                            annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                            aaa.coordinate=annoo;
                            aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;
                            aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                            aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                            [_mapview addAnnotation:aaa];

                        }
                }
            }
        }
    }
}

```

```

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

break;

case 1:
{
    tag=2;

    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"economic"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

                    aaa.coordinate=annoo;
                    aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;
                    aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                    aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                    [_mapview addAnnotation:aaa];

                }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

break;

```



```

case 2:
{
    tag=3;
    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"social"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

                    aaa.coordinate=annoo;
                    aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;
                    aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                    aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                    [_mapview addAnnotation:aaa];

                }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

break;

case 3:
{
    tag=4;
    [_mapview removeAnnotations:[_mapview annotations]];
    PFQuery *query0 = [PFQuery queryWithClassName:@"EditMap"];
    [query0 whereKey:@"pillar" equalTo:@"cultural"];
    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longitude"] floatValue];

```

```

        aaa.coordinate=annoo;
        aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;
        aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
        aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

        [_mapview addAnnotation:aaa];
    }

    } else {
        // Log details of the failure
        NSLog(@"Error: %@ %@", error, [error userInfo]);
    }

}];

}

break;

case 4:
{
    tag=0;
    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query1 = [PFQuery queryWithClassName:@"EditMap"];

    [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                    annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                    annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                    aaa.coordinate=annoo;
                    aaa.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname"] ;
                    aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                    aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                    [_mapview addAnnotation:aaa];

                }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }
    }
}

```

```

    }
    break;
case UISegmentedControlNoSegment:
    break;
default:
    NSLog(@"No option for: %d", [segmentedControl selectedSegmentIndex]);
}

}

- (IBAction)fiveplaces:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;
    switch ([segmentedControl selectedSegmentIndex]) {
        case 0:
            [_mapview removeOverlays:[_mapview overlays]];

            CLLocationCoordinate2D coords[9]={
                CLLocationCoordinate2DMake(45.507883, -61.042528),
                CLLocationCoordinate2DMake(45.510199, -61.041669),
                CLLocationCoordinate2DMake(45.510981, -61.033859),
                CLLocationCoordinate2DMake(45.513206, -61.028537),
                CLLocationCoordinate2DMake(45.516394, -60.994677),
                CLLocationCoordinate2DMake(45.512635, -60.990471),
                CLLocationCoordinate2DMake(45.510379, -60.998368),
                CLLocationCoordinate2DMake(45.511733, -61.001758),
                CLLocationCoordinate2DMake(45.507883, -61.042528),
            };

            MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:9];
            [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.512996,-
61.014161)];
            [_mapview addOverlay:polyLine];
            MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
            [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.512996,-61.014161), span)
animated:YES];
            self.arenaInfo.text=@"Arichat Waterfront:High volume of historical photos
available due to its importance in the past Populated area on Isle Madame Highly utilized
area Area of noticeable erosion Important infrastructure: Substantial seafood industry
holdings in this area (Government wharf, Premium Seafood Inc. wharf and infrastructure),
town sewer facilities.";
        }
        break;
        case 1:
        {
            [_mapview removeOverlays:[_mapview overlays]];

            CLLocationCoordinate2D coords[10]={
                CLLocationCoordinate2DMake(45.514048, -60.956010),
                CLLocationCoordinate2DMake(45.512725, -60.961461),
                CLLocationCoordinate2DMake(45.510259, -60.964980),
                CLLocationCoordinate2DMake(45.503492, -60.968156),
                CLLocationCoordinate2DMake(45.498950, -60.969915),
                CLLocationCoordinate2DMake(45.494047, -60.959401),
            };
        }
    }
}

```

```

        CLLocationCoordinate2DMake(45.498860, -60.954508),
        CLLocationCoordinate2DMake(45.504635, -60.956053),
        CLLocationCoordinate2DMake(45.512214, -60.952148),
        CLLocationCoordinate2DMake(45.514048, -60.956010),

};

MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:10];
[_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.505898,-
60.960130)];
[_mapview addOverlay:polyLine];
MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
[_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.505898,-60.960130), span)
animated:YES];
    self.areasinfor.text=@"Petit de Grat Packers:Area of noticeable impact- Highly
vulnerable Economic impact to local industry: Important seasonal source of income for
many community members (Approximately 90 employees) as well as selling opportunities for
crab fishermen.";

}
break;
case 2:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[11]={
        CLLocationCoordinate2DMake(45.506319, -60.944509),
        CLLocationCoordinate2DMake(45.502710, -60.940990),
        CLLocationCoordinate2DMake(45.500845, -60.936355),
        CLLocationCoordinate2DMake(45.496032, -60.935497),
        CLLocationCoordinate2DMake(45.494588, -60.938158),
        CLLocationCoordinate2DMake(45.492693, -60.939531),
        CLLocationCoordinate2DMake(45.491038, -60.933437),
        CLLocationCoordinate2DMake(45.493264, -60.929660),
        CLLocationCoordinate2DMake(45.500664, -60.932750),
        CLLocationCoordinate2DMake(45.503642, -60.933523),
        CLLocationCoordinate2DMake(45.506319, -60.944509),

    };

    MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:11];
    [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.498288,-
60.933995)];
    [_mapview addOverlay:polyLine];
    MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
    [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.498288,-60.933995), span)
animated:YES];
    self.areasinfor.text=@"The Little Anse Breakwater:Highly vulnerable area
Noticeable erosion Residents have experienced isolation through impassable highways
Houses are threatened by sea level rise.";

}
break;
case 3:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[7]={
        CLLocationCoordinate2DMake(45.607165, -61.088147),
        CLLocationCoordinate2DMake(45.586325, -61.094584),
        CLLocationCoordinate2DMake(45.576113, -61.069350),
        CLLocationCoordinate2DMake(45.582181, -61.038708),
        CLLocationCoordinate2DMake(45.601580, -61.044545),
        CLLocationCoordinate2DMake(45.608126, -61.059394),
        CLLocationCoordinate2DMake(45.607165, -61.088147),

```

```

};

MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:7];
[_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.593773,-
61.067290)];
[_mapview addOverlay:polyLine];
MKCoordinateSpan span = MKCoordinateSpanMake(0.1, 0.1);
[_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.593773,-61.067290), span)
animated:YES];
    self.arenaInfo.text=@"Lennox Passage Bridge:Critical access to the island:
Only entering and exiting point for the entire island for inhabitants. Emergency access,
tourism access etc..";

}
    break;
case 4:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords[11]={
        CLLocationCoordinate2DMake(45.514680, -61.091837),
        CLLocationCoordinate2DMake(45.517747, -61.090164),
        CLLocationCoordinate2DMake(45.516935, -61.084199),
        CLLocationCoordinate2DMake(45.520243, -61.082911),
        CLLocationCoordinate2DMake(45.511101, -61.069178),
        CLLocationCoordinate2DMake(45.509657, -61.070037),
        CLLocationCoordinate2DMake(45.511131, -61.076088),
        CLLocationCoordinate2DMake(45.513507, -61.080207),
        CLLocationCoordinate2DMake(45.514650, -61.083083),
        CLLocationCoordinate2DMake(45.513056, -61.085186),
        CLLocationCoordinate2DMake(45.514680, -61.091837),

    };

    MKPolyline *polyLine=[MKPolyline polylineWithCoordinates:coords count:11];
    [_mapview setCenterCoordinate:CLLocationCoordinate2DMake(45.515552,-
61.079950)];
    [_mapview addOverlay:polyLine];
    MKCoordinateSpan span = MKCoordinateSpanMake(0.05, 0.05);
    [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.515552,-61.079950), span)
animated:YES];
    self.arenaInfo.text=@"Creighton Island:Area of noticeable impact: The area
has been cut off forcing inhabitants to relocate off the island within a small amount of
time.";

}
    break;
case 5:
{
    [_mapview removeOverlays:[_mapview overlays]];

    CLLocationCoordinate2D coords1[9]={
        CLLocationCoordinate2DMake(45.507883, -61.042528),
        CLLocationCoordinate2DMake(45.510199, -61.041669),
        CLLocationCoordinate2DMake(45.510981, -61.033859),
        CLLocationCoordinate2DMake(45.513206, -61.028537),
        CLLocationCoordinate2DMake(45.516394, -60.994677),
        CLLocationCoordinate2DMake(45.512635, -60.990471),
        CLLocationCoordinate2DMake(45.510379, -60.998368),
        CLLocationCoordinate2DMake(45.511733, -61.001758),
        CLLocationCoordinate2DMake(45.507883, -61.042528),

    };

    MKPolyline *polyLine1=[MKPolyline polylineWithCoordinates:coords1 count:9];
    [_mapview addOverlay:polyLine1];

```

```

CLLocationCoordinate2D coords2[10]={
    CLLocationCoordinate2DMake(45.514048, -60.956010),
    CLLocationCoordinate2DMake(45.512725, -60.961461),
    CLLocationCoordinate2DMake(45.510259, -60.964980),
    CLLocationCoordinate2DMake(45.503492, -60.968156),
    CLLocationCoordinate2DMake(45.498950, -60.969915),
    CLLocationCoordinate2DMake(45.494047, -60.959401),
    CLLocationCoordinate2DMake(45.498860, -60.954508),
    CLLocationCoordinate2DMake(45.504635, -60.956053),
    CLLocationCoordinate2DMake(45.512214, -60.952148),
    CLLocationCoordinate2DMake(45.514048, -60.956010),
};

MKPolyline *polyLine2=[MKPolyline polylineWithCoordinates:coords2 count:10];
[_mapview addOverlay:polyLine2];

CLLocationCoordinate2D coords3[11]={
    CLLocationCoordinate2DMake(45.506319, -60.944509),
    CLLocationCoordinate2DMake(45.502710, -60.940990),
    CLLocationCoordinate2DMake(45.500845, -60.936355),
    CLLocationCoordinate2DMake(45.496032, -60.935497),
    CLLocationCoordinate2DMake(45.494588, -60.938158),
    CLLocationCoordinate2DMake(45.492693, -60.939531),
    CLLocationCoordinate2DMake(45.491038, -60.933437),
    CLLocationCoordinate2DMake(45.493264, -60.929660),
    CLLocationCoordinate2DMake(45.500664, -60.932750),
    CLLocationCoordinate2DMake(45.503642, -60.933523),
    CLLocationCoordinate2DMake(45.506319, -60.944509),
};

MKPolyline *polyLine3=[MKPolyline polylineWithCoordinates:coords3 count:11];
[_mapview addOverlay:polyLine3];

CLLocationCoordinate2D coords4[7]={
    CLLocationCoordinate2DMake(45.607165, -61.088147),
    CLLocationCoordinate2DMake(45.586325, -61.094584),
    CLLocationCoordinate2DMake(45.576113, -61.069350),
    CLLocationCoordinate2DMake(45.582181, -61.038708),
    CLLocationCoordinate2DMake(45.601580, -61.044545),
    CLLocationCoordinate2DMake(45.608126, -61.059394),
    CLLocationCoordinate2DMake(45.607165, -61.088147),
};

MKPolyline *polyLine4=[MKPolyline polylineWithCoordinates:coords4 count:7];
[_mapview addOverlay:polyLine4];

CLLocationCoordinate2D coords5[11]={
    CLLocationCoordinate2DMake(45.514680, -61.091837),
    CLLocationCoordinate2DMake(45.517747, -61.090164),
    CLLocationCoordinate2DMake(45.516935, -61.084199),
    CLLocationCoordinate2DMake(45.520243, -61.082911),
    CLLocationCoordinate2DMake(45.511101, -61.069178),
    CLLocationCoordinate2DMake(45.509657, -61.070037),
    CLLocationCoordinate2DMake(45.511131, -61.076088),
    CLLocationCoordinate2DMake(45.513507, -61.080207),
    CLLocationCoordinate2DMake(45.514650, -61.083083),
    CLLocationCoordinate2DMake(45.513056, -61.085186),
    CLLocationCoordinate2DMake(45.514680, -61.091837),
};

MKPolyline *polyLine5=[MKPolyline polylineWithCoordinates:coords5 count:11];

```

```

        [_mapview addOverlay:polyLine5];
        self.areaInfo.text=nil;

        MKCoordinateSpan span = MKCoordinateSpanMake(0.18, 0.18);
        [_mapview
setRegion:MKCoordinateRegionMake(CLLocationCoordinate2DMake(45.550815,-61.023674), span)
animated:YES];
    }

    break;
case UISegmentedControlNoSegment:
    // do something

    break;
default:
    NSLog(@"No option for: %d", [segmentedControl selectedIndex]);
}

}

- (IBAction)areasandplaces:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;

    switch([segmentedControl selectedIndex])
    {
        case 0:
        {
            PFQuery *query2 = [PFQuery queryWithClassName:@"Dangersareas"];

            [query2 findObjectsInBackgroundWithBlock:^(NSArray *objects, NSError *error)
{
                if (!error) {

                    for (int i=0; i<objects.count; i++) {

                        NSArray *ff;
                        ff= [ [objects objectAtIndex:i] objectForKey:@"latitude"];

                        NSArray *hh;
                        hh= [ [objects objectAtIndex:i] objectForKey:@"longtitude"];

                        apoint =[[ [objects objectAtIndex:i] objectForKey:@"index"]
integerValue];

                        for (int i=0; i<apoint; i++) {
                            coordss[i].latitude = [[ff objectAtIndex:i] floatValue];
                        }
                        for (int i=0; i<apoint; i++) {
                            coordss[i].longitude=[[hh objectAtIndex:i] floatValue ];
                        }
                        MKPolygon *polygon;
                        polygon = [MKPolygon polygonWithCoordinates:coordss count:apoint];

                        [_mapview addOverlay:polygon];

                        CLLocationCoordinate2D points[2];
                        points[0]= CLLocationCoordinate2DMake(coordss[0].latitude,
coordss[0].longitude);

```

```

        points[1]= CLLocationCoordinate2DMake(coordss[apoint-1].latitude,
coordss[apoint-1].longitude);
        MKPolyline *polyline;
        polyline = [MKPolyline polylineWithCoordinates:points count:2];
        [_mapview addOverlay:polyline];
    }
    } else {
        // Log details of the failure
        NSLog(@"Error: %@ %@", error, [error userInfo]);
    }
    }];

}

break;
case 1:
{

    tag=5;
    PFQuery *query2 = [PFQuery queryWithClassName:@"Dangersplaces"];

    [query2 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
    {
        if (!error) {

            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

            { MyAnnotation *anot=[[MyAnnotation alloc]init];

                annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                anot.coordinate=annoo;
                anot.title=[ [objectss objectAtIndex:i] objectForKey:@"title"];
                anot.subtitle=[ [objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                anot.iconname=[ [objectss objectAtIndex:i]
objectForKey:@"iconname" ] ;

                [_mapview addAnnotation:anot];

            }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }
    }];
}

```



```

        }
        break;

        default:
            break;
    }

}

- (IBAction)helpandhelper:(UISegmentedControl *)sender {
    UISegmentedControl *segmentedControl = (UISegmentedControl*) sender;
    switch([segmentedControl selectedSegmentIndex]){
        case 0:{
            tag=6;
            [_mapview removeAnnotations:[_mapview annotations]];

            PFQuery *query1 = [PFQuery queryWithClassName:@"share"];
            [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error)
            {
                if (!error) {
                    CLLocationCoordinate2D annoo;
                    iconnumber=objectss.count;
                    for (int i=0; i<objectss.count; i++)

                        { MyAnnotation *aaa=[[MyAnnotation alloc]init];

                            annoo.latitude=[[ [objectss objectAtIndex:i]
objectForKey:@"latitude"] floatValue ];
                            annoo.longitude=[[ [objectss objectAtIndex:i]
objectForKey:@"longtitude"] floatValue];

                            aaa.coordinate=annoo;
                            aaa.iconname=[ [objectss objectAtIndex:i] objectForKey:@"icon" ] ;
                            aaa.subtitle=[[objectss objectAtIndex:i]
objectForKey:@"subtitle"];
                            aaa.title=[[objectss objectAtIndex:i] objectForKey:@"title"];

                            [_mapview addAnnotation:aaa];

                        }

                } else {
                    // Log details of the failure
                    NSLog(@"Error: %@ %@", error, [error userInfo]);
                }

            }];

        }
        break;
        case 1:{
        }
        break;
    }
}

```

```

        default:
            break;
    }

}

- (void)viewDidLoad
{
    [super viewDidLoad];
    fourpillars.hidden=YES;
    fiveplaces.hidden=YES;
    areasandplaces.hidden=YES;
    helpandhelper.hidden=YES;
}

-(void)viewDidAppear:(BOOL)animated
{
    _mapview.showsUserLocation = YES;
    _mapview.zoomEnabled = YES;
    _mapview.scrollEnabled = YES;
    _mapview.mapType = MKMapTypeStandard;
    _mapview.delegate=self;

    CLLocationCoordinate2D p;
    p.latitude=45.550815;
    p.longitude=-61.023674;
    MKCoordinateRegion mapRegion;
    mapRegion.center = p;
    mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    [_mapview setRegion:mapRegion animated: YES];

    PFQuery *query0 = [PFQuery queryWithClassName:@"share"];

    [query0 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error) {
        if (!error) {

            _numberofusers.text=[NSString stringWithFormat:@"%d",objectss.count];

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }
    }];
}

- (void)mapView:(MKMapView *)mapView didUpdateUserLocation:(MKUserLocation *)userLocation
{
    if(haveAlreadyReceivedCoordinates) {
        return;
    }

    haveAlreadyReceivedCoordinates = YES;
    CLLocationCoordinate2D p;
    p.latitude=45.550815;
    p.longitude=-61.023674;
    MKCoordinateRegion mapRegion;
    mapRegion.center = p;
    mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    [_mapview setRegion:mapRegion animated: YES];
}

```

```

- (MKOverlayView *)mapView:(MKMapView *)mapView viewForOverlay:(id)overlay
{
    /*MKPolygonView *polygonView ;
    polygonView.lineWidth = 0.5;
    polygonView.strokeColor = [UIColor redColor];
    polygonView.fillColor = [[UIColor colorWithRed:0.9176 green:0.9098 blue:0.8117
alpha:1.0] colorWithAlphaComponent:0.5];

    return polygonView;*/

    MKPolylineView *polyLineView = [[MKPolylineView alloc] initWithOverlay: overlay];
    polyLineView.strokeColor = [UIColor blueColor];
    polyLineView.lineWidth = 4.0;

    //polyLineView.backgroundColor= [UIColor blueColor];
    //polyLineView.alpha=0.3f;

    return polyLineView;
}

- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
<MKAnnotation>)Annotation
{
    MyAnnotation *annoo=(MyAnnotation *)Annotation;

    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    if(Annotation != _mapview.userLocation)
    {
        for (int i=0; i<iconnumber; i++) {
            if (tag==0)
            {
                NSString * name=annoo.iconname;
                NSString *png=@"png";
                NSString *all=[name stringByAppendingString:png];
                UIImage * img =[UIImage imageNamed:all];
                CGRect resizeRect;
                resizeRect.size.height = 20;
                resizeRect.size.width = 20;
                resizeRect.origin = (CGPoint){0.0f, 0.0f};
                UIGraphicsBeginImageContext(resizeRect.size);
                [img drawInRect:resizeRect];
                UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
                UIGraphicsEndImageContext();
                custview.image = resizedImage;
                custview.canShowCallout=YES;
                custview.enabled = YES;
                custview.multipleTouchEnabled = NO;
                return custview;
            }else if (tag==5)
            {
                NSString * name=annoo.iconname;

```

```

NSString *png=@"png";
NSString *all=[name stringByAppendingString:png];
UIImage *img=[UIImage imageNamed:all];
CGRect resizeRect;
resizeRect.size.height = 20;
resizeRect.size.width = 20;
resizeRect.origin = (CGPoint){0.0f, 0.0f};
UIGraphicsBeginImageContext(resizeRect.size);
[img drawInRect:resizeRect];
UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
UIGraphicsEndImageContext();
custview.image = resizedImage;
custview.canShowCallout=YES;
custview.enabled = YES;
custview.multipleTouchEnabled = NO;
return custview;
}

else if (tag==1)
{
    NSString * name=annoo.iconname;
    NSString *png=@"png";
    NSString *all=[name stringByAppendingString:png];
    UIImage *img=[UIImage imageNamed:all];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

else if (tag==2)
{
    NSString * name=annoo.iconname;
    NSString *png=@"png";
    NSString *all=[name stringByAppendingString:png];
    UIImage *img=[UIImage imageNamed:all];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

else if (tag==3)
{
    NSString * name=annoo.iconname;
    NSString *png=@"png";

```

```

        NSString *all=[name stringByAppendingString:png];
        UIImage *img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==4)
    {
        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage *img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if (tag==6)
    {
        NSString * name=annoo.iconname;
        NSString *png=@"png";
        NSString *all=[name stringByAppendingString:png];
        UIImage *img =[UIImage imageNamed:all];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

    }

}
else
    [_mapview.userLocation setTitle:@"I am here"];
return nil;

```

```

}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end

```

MapEditView

```

#import <UIKit/UIKit.h>
#import<MapKit/MapKit.h>
@interface MapEditViewController : UIViewController<MKMapViewDelegate,
UITextFieldDelegate>
@property (strong, nonatomic) IBOutlet MKMapView *mapview;
@property (strong, nonatomic) IBOutlet UITextField *annotitle;
@property (strong, nonatomic) IBOutlet UITextField *annosubtitle;

@end

#import "MapEditViewController.h"
#import<MapKit/MapKit.h>
#import <Parse/Parse.h>
#import <UIKit/UIKit.h>

CLLocationCoordinate2D coord;
NSString *iconname;
NSString *pillar;
int a;

@interface MapEditViewController ()

@end

@implementation MapEditViewController
@synthesize mapview=_mapview;
@synthesize annotitle=_annotitle;
@synthesize annosubtitle=_annosubtitle;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (IBAction)cancel:(id)sender {
    [self dismissViewControllerAnimated:YES completion:nil];
    iconname=nil;
}

```

```

- (IBAction)done:(id)sender {
    if ([self.annotitle.text length] != 0 || [self.annosubtitle.text length] != 0) {
        PFObject *placeObject = [PFObject objectWithClassName:@"EditMap"];

        NSString *index=iconname;
        NSString *iconpillar=pillar;

        NSNumber *latitude = [NSNumber numberWithFloat:coord.latitude];

        NSNumber *longitude = [NSNumber numberWithFloat:coord.longitude];

        NSString *title=self.annotitle.text;
        NSString *subtitle=self.annosubtitle.text;
        [placeObject setObject: title forKey:@"title"];

        [placeObject setObject: subtitle forKey:@"subtitle"];

        [placeObject setObject: latitude forKey:@"latitude"];

        [placeObject setObject: longitude forKey:@"longitude"];

        [placeObject setObject: index forKey:@"iconname"];

        [placeObject setObject: iconpillar forKey:@"pillar"];
        [placeObject saveInBackground];

        [self dismissViewControllerAnimated:YES completion:nil];

    }else{

        UIAlertView *message = [[UIAlertView alloc] initWithTitle:@"Alert"
                                                                message:@"Step One: Longpress
map to add icon. Step Two: Choose the type of icon. Step Three: Inpt icon title and
subtitle. Step Four: Press done to finish."
                                                                delegate:nil
                                                                cancelButtonTitle:@"OK"
                                                                otherButtonTitles:nil];

        [message show];

    }
}

- (IBAction)coffee:(UIButton *)sender {

    iconname=@"coffee";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"economic";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}

```

```

}

- (IBAction)airport:(UIButton *)sender {

    iconname=@"airport";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"social";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}

- (IBAction)bus:(UIButton *)sender {
    iconname=@"bus";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"social";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}

- (IBAction)church:(UIButton *)sender {
    iconname=@"church";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"cultural";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}

- (IBAction)site:(UIButton *)sender {
    iconname=@"site";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

```



```

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"cultural";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }

    - (IBAction)beach:(UIButton *)sender {
        iconname=@"beach";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"environmental";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }

    - (IBAction)gas:(UIButton *)sender {
        iconname=@"gas";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"economic";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }

    - (IBAction)grocery:(UIButton *)sender {
        iconname=@"grocery";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;

```

```

        pillar=@"economic";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }
    - (IBAction)hall:(UIButton *)sender {
        iconname=@"hall";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"social";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }
    - (IBAction)hospital:(UIButton *)sender {
        iconname=@"hospital";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"social";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }
    - (IBAction)house:(UIButton *)sender {
        iconname=@"house";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

        NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
        NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
        NSString *coordinate= [la stringByAppendingString:lo];

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"social";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }
    - (IBAction)police:(UIButton *)sender {
        iconname=@"police";

        MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

        point.coordinate = coord;

```

```

NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
NSString *coordinate= [la stringByAppendingString:lo];

point.title = @"Coordinate";
point.subtitle=coordinate;
pillar=@"social";
[_mapview removeAnnotations: _mapview.annotations] ;

[self.mapview addAnnotation:point];
}
- (IBAction)bank:(UIButton *)sender {
    iconname=@"bank";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"economic";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
- (IBAction)bike:(UIButton *)sender {
    iconname=@"bike";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"environmental";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
- (IBAction)trail:(UIButton *)sender {
    iconname=@"trail";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"environmental";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
}

```

```

- (IBAction)port:(UIButton *)sender {
    iconname=@"port";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"social";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
- (IBAction)railway:(UIButton *)sender {
    iconname=@"railway";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"social";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
- (IBAction)restaurant:(UIButton *)sender {
    iconname=@"restaurant";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    pillar=@"economic";
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}
- (IBAction)tree:(UIButton *)sender {
    iconname=@"tree";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

```

```

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        pillar=@"environmental";
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
    }

- (void)viewDidLoad
{
    [super viewDidLoad];

    UILongPressGestureRecognizer *lpgr = [[UILongPressGestureRecognizer alloc]
                                           initWithTarget:self
                                           action:@selector(handleLongPress:)];
    lpgr.minimumPressDuration = 0.5; //user needs to press for 2 seconds
    [self.mapview addGestureRecognizer:lpgr];
}

- (void)viewDidAppear:(BOOL)animated
{
    _mapview.showsUserLocation = YES;
    _mapview.zoomEnabled = YES;
    _mapview.scrollEnabled = YES;
    _mapview.mapType = MKMapTypeStandard;
    _mapview.delegate=self;

    CLLocationCoordinate2D p;
    p.latitude=45.550815;
    p.longitude=-61.023674;
    MKCoordinateRegion mapRegion;
    mapRegion.center = p;
    mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    [_mapview setRegion:mapRegion animated: YES];
}

- (void)handleLongPress:(UIGestureRecognizer *)gestureRecognizer
{
    if (gestureRecognizer.state != UIGestureRecognizerStateBegan)
        return;

    CGPoint touchPoint = [gestureRecognizer locationInView:self.mapview];

    CLLocationCoordinate2D touchMapCoordinate =
    [self.mapview convertPoint:touchPoint toCoordinateFromView:self.mapview];

    coord=touchMapCoordinate;

    NSString *la= [NSString stringWithFormat:@"%f",touchMapCoordinate.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",touchMapCoordinate.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = touchMapCoordinate;
    point.title = @"Coordinate";
    point.subtitle = coordinate;

    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
}

```

```

}

- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id)annotation
{
    MKPinAnnotationView *pinView = nil;

    if(annotation != _mapview.userLocation)
    {
        if (iconname==nil)
        {
            static NSString *defaultPinID = @"com.invasivecode.pin";

            pinView = (MKPinAnnotationView *)[_mapview
            dequeueReusableAnnotationViewWithIdentifier:defaultPinID];
            if ( pinView == nil )
                pinView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
                reuseIdentifier:@"currentloc"];
            pinView.pinColor = MKPinAnnotationColorPurple;
            pinView.canShowCallout = YES;
            pinView.animatesDrop = YES;

        }

        else if ([iconname isEqual:@"coffee"])
        {
            MKAnnotationView *custview=[[MKAnnotationView alloc] init];

            UIImage * img =[UIImage imageNamed:@"coffee.png"];
            CGRect resizeRect;
            resizeRect.size.height = 20;
            resizeRect.size.width = 20;
            resizeRect.origin = (CGPoint){0.0f, 0.0f};
            UIGraphicsBeginImageContext(resizeRect.size);
            [img drawInRect:resizeRect];
            UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
            UIGraphicsEndImageContext();
            custview.image = resizedImage;
            custview.canShowCallout=YES;
            custview.enabled = YES;
            custview.multipleTouchEnabled = NO;
            return custview;
        }
        else if ([iconname isEqual:@"airport"])
        {
            MKAnnotationView *custview=[[MKAnnotationView alloc] init];

            UIImage * img =[UIImage imageNamed:@"airport.png"];
            CGRect resizeRect;
            resizeRect.size.height = 20;
            resizeRect.size.width = 20;
            resizeRect.origin = (CGPoint){0.0f, 0.0f};
            UIGraphicsBeginImageContext(resizeRect.size);
            [img drawInRect:resizeRect];

```

```

UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
UIGraphicsEndImageContext();
custview.image = resizedImage;
custview.canShowCallout=YES;
custview.enabled = YES;
custview.multipleTouchEnabled = NO;
return custview;
}else if ([iconname isEqual:@"beach"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"beach.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;

    return custview;
}else if ([iconname isEqual:@"bus"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];
    UIImage * img =[UIImage imageNamed:@"bus.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;

    return custview;
}else if ([iconname isEqual:@"church"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"church.png"];
    CGRect resizeRect;
    resizeRect.size.height = 32;
    resizeRect.size.width = 32;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;

```

```

        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if ([iconname isEqual:@"site"])
    {

        MKAnnotationView *custview=[[MKAnnotationView alloc] init];

        UIImage * img =[UIImage imageNamed:@"site.png"];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if ([iconname isEqual:@"gas"])
    {

        MKAnnotationView *custview=[[MKAnnotationView alloc] init];

        UIImage * img =[UIImage imageNamed:@"gas.png"];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if ([iconname isEqual:@"grocery"])
    {

        MKAnnotationView *custview=[[MKAnnotationView alloc] init];

        UIImage * img =[UIImage imageNamed:@"grocery.png"];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

```



```

}else if ([iconname isEqual:@"hall"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"hall.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

}else if ([iconname isEqual:@"hospital"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"hospital.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

}else if ([iconname isEqual:@"house"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"house.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

}else if ([iconname isEqual:@"police"])
{

```

```

MKAnnotationView *custview=[[MKAnnotationView alloc] init];

UIImage * img =[UIImage imageNamed:@"police.png"];
CGRect resizeRect;
resizeRect.size.height = 20;
resizeRect.size.width = 20;
resizeRect.origin = (CGPoint){0.0f, 0.0f};
UIGraphicsBeginImageContext(resizeRect.size);
[img drawInRect:resizeRect];
UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
UIGraphicsEndImageContext();
custview.image = resizedImage;
custview.canShowCallout=YES;
custview.enabled = YES;
custview.multipleTouchEnabled = NO;
return custview;
}
else if ([iconname isEqual:@"bank"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"bank.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}
else if ([iconname isEqual:@"bike"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"bike"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;

    return custview;
}
else if ([iconname isEqual:@"trail"])
{
    MKAnnotationView *custview=[[MKAnnotationView alloc] init];

```

```

UIImage * img =[UIImage imageNamed:@"trail"];
CGRect resizeRect;
resizeRect.size.height = 20;
resizeRect.size.width = 20;
resizeRect.origin = (CGPoint){0.0f, 0.0f};
UIGraphicsBeginImageContext(resizeRect.size);
[img drawInRect:resizeRect];
UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
UIGraphicsEndImageContext();
custview.image = resizedImage;
custview.canShowCallout=YES;
custview.enabled = YES;
custview.multipleTouchEnabled = NO;
return custview;
}

}else if ([iconname isEqual:@"port"])
{

    MKAnnotationView *custview=[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"port.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

}else if ([iconname isEqual:@"railway"])
{

    MKAnnotationView *custview=[MKAnnotationView alloc] init];

    UIImage * img =[UIImage imageNamed:@"railway.png"];
    CGRect resizeRect;
    resizeRect.size.height = 20;
    resizeRect.size.width = 20;
    resizeRect.origin = (CGPoint){0.0f, 0.0f};
    UIGraphicsBeginImageContext(resizeRect.size);
    [img drawInRect:resizeRect];
    UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
    UIGraphicsEndImageContext();
    custview.image = resizedImage;
    custview.canShowCallout=YES;
    custview.enabled = YES;
    custview.multipleTouchEnabled = NO;
    return custview;
}

}else if ([iconname isEqual:@"restaurant"])
{

    MKAnnotationView *custview=[MKAnnotationView alloc] init];

```

```

        UIImage * img =[UIImage imageNamed:@"restaurant.png"];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if ([iconname isEqual:@"tree"])
    {

        MKAnnotationView *custview=[[MKAnnotationView alloc] init];

        UIImage * img =[UIImage imageNamed:@"tree.png"];
        CGRect resizeRect;
        resizeRect.size.height = 20;
        resizeRect.size.width = 20;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

    }
    else
        [_mapview.userLocation setTitle:@"I am here"];

    return pinView;
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end

```

AddDangerPlace

```
#import <UIKit/UIKit.h>
```

```

#import<MapKit/MapKit.h>
#import <MobileCoreServices/MobileCoreServices.h>

@interface AddDangersPlaceViewController : UIViewController<MKMapViewDelegate,
UITextFieldDelegate,UITableViewDelegate,
UITableViewDataSource,UINavigationControllerDelegate>

@property (strong, nonatomic) IBOutlet MKMapView *mapview;
@property (strong, nonatomic) IBOutlet UITextField *placenotification;
@property (strong, nonatomic) IBOutlet UITableView *table;
@property (strong, nonatomic) IBOutlet UIBarButtonItem *cancel;
@property (strong, nonatomic) IBOutlet UIButton *red;
@property (strong, nonatomic) IBOutlet UIButton *yellow;
@property (strong, nonatomic) IBOutlet UIButton *green;
@property (strong, nonatomic) IBOutlet UITextField *annotitle;
@property (strong, nonatomic) IBOutlet UITextField *annosubtitle;
@property (strong, nonatomic) IBOutlet UILabel *tablenotification;
@property (strong, nonatomic) IBOutlet UILabel *tabletitle;
@property (strong, nonatomic) IBOutlet UILabel *tablesubtitle;
@property (strong, nonatomic) IBOutlet UISegmentedControl *maptype;

@end

#import "AddDangersPlaceViewController.h"
#import<MapKit/MapKit.h>
#import <Parse/Parse.h>
#import <UIKit/UIKit.h>
#import "MyAnnotation.h"

CLLocationCoordinate2D coord;
NSString *iconname;
int a;
NSString *name;
int iconnumber;
NSString *ppppp;
static BOOL haveAlreadyReceivedCoordinates = NO;

@interface AddDangersPlaceViewController ()

@end

@implementation AddDangersPlaceViewController
{
    NSArray *tableData;
    NSUserDefaults *tabledefault;
    NSMutableArray *myList;
    NSMutableArray *detail;

}

@synthesize mapview=_mapview;
@synthesize placenotification=_placenotification;
@synthesize table=_table;
@synthesize cancel;
@synthesize red;
@synthesize yellow;
@synthesize green;
@synthesize annotitle=_annotitle;
@synthesize annosubtitle=_annosubtitle;
@synthesize tablenotification=_tablenotification;
@synthesize tabletitle=_tabletitle;
@synthesize tablesubtitle=_tablesubtitle;
@synthesize maptype;

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil

```

```

{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (IBAction)cancel:(UIBarButtonItem *)sender {

    [self dismissViewControllerAnimated:YES completion:nil];

    iconname=nil;
}
- (IBAction)confirm:(UIBarButtonItem *)sender {
    if ([self.placenotification.text length]!=0 && [iconname length]!=0
    &&[self.annotitle.text length]!=0 && [self.annosubtitle.text length]!=0) {
        PFObject *placeObject = [PFObject objectWithClassName:@"Dangersplaces"];
        PFObject *placeObject2 = [PFObject objectWithClassName:@"notification"];
        NSString *index=iconname;

        NSNumber *latitude = [NSNumber numberWithFloat:coord.latitude];

        NSNumber *longitude = [NSNumber numberWithFloat:coord.longitude];

        NSString *notification=self.placenotification.text;
        NSString *atitle=self.annotitle.text;
        NSString *asubtitle=self.annosubtitle.text;

        NSDateFormatter *dateForm = [[NSDateFormatter alloc] init];
        [dateForm setDateFormat:@"yyyy-MM-dd hh:mm:ss"];
        NSString *dateString = [dateForm stringFromDate:[NSDate date]];

        [placeObject setObject: notification forKey:@"notification"];
        [placeObject2 setObject: notification forKey:@"place"];
        [placeObject2 setObject: @" " forKey:@"area"];

        [placeObject setObject: atitle forKey:@"title"];
        [placeObject2 setObject: asubtitle forKey:@"subtitle"];

        [placeObject setObject: latitude forKey:@"latitude"];

        [placeObject2 setObject: longitude forKey:@"longitude"];

        [placeObject setObject: index forKey:@"iconname"];

        [placeObject2 setObject: dateString forKey:@"date"];

        [placeObject saveInBackground];
        [placeObject2 saveInBackground];
        [self dismissViewControllerAnimated:YES completion:nil];

    }else{

        UIAlertView *message = [[UIAlertView alloc] initWithTitle:@"Check Steps"

```

```

message:@"Step 1: Press plus
sign to enable map edit. Step 2; Longpress map to add icons. Step 3: Choose the type of
icon. Step 4: Input required textfiled. Step 5: Press done to submit."
delegate:nil
cancelButtonTitle:@"OK"
otherButtonTitles:nil];

```

```

[message show];

```

```

    }
}

```

```

- (IBAction)redalert:(UIButton *)sender {
    iconname=@"red";
    name=@"start";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
    name=nil;
}

```

```

- (IBAction)yellowalert:(UIButton *)sender {
    iconname=@"yellow";
    name=@"start";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    point.title = @"Coordinate";
    point.subtitle=coordinate;
    [_mapview removeAnnotations: _mapview.annotations] ;

    [self.mapview addAnnotation:point];
    name=nil;
}

```

```

- (IBAction)greenalert:(UIButton *)sender {
    iconname=@"green";
    name=@"start";

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = coord;

    NSString *la= [NSString stringWithFormat:@"%f",coord.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",coord.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

```

```

        point.title = @"Coordinate";
        point.subtitle=coordinate;
        [_mapview removeAnnotations: _mapview.annotations] ;

        [self.mapview addAnnotation:point];
        name=nil;
    }

- (void)viewDidLoad
{
    [super viewDidLoad];

    [self.table setDelegate:self];

    [self.table setDataSource:self];
    _mapview.delegate=self;

    name=@"start";
    red.hidden=YES;
    yellow.hidden=YES;
    green.hidden=YES;

    UILongPressGestureRecognizer *lpgr = [[UILongPressGestureRecognizer alloc]
                                           initWithTarget:self
                                           action:@selector(handleLongPress:)];
    lpgr.minimumPressDuration = 0.5; //user needs to press for 2 seconds
    [self.mapview addGestureRecognizer:lpgr];

    iconname=nil;

    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query1 = [PFQuery queryWithClassName:@"Dangersplaces"];

    [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error) {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)
            {
                MyAnnotation *aaa=[[MyAnnotation alloc]init];

                annoo.latitude=[[ [objectss objectAtIndex:i] objectForKey:@"latitude"]
floatValue ];
                annoo.longitude=[[ [objectss objectAtIndex:i] objectForKey:@"longtitude"]
floatValue];

                aaa.coordinate=annoo;
                aaa.title=[ [objectss objectAtIndex:i] objectForKey:@"title"];
                aaa.subtitle=[ [objectss objectAtIndex:i] objectForKey:@"subtitle"];
                aaa.iconname=[ [objectss objectAtIndex:i] objectForKey:@"iconname"] ;

                [_mapview addAnnotation:aaa];
            }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }
    }
}

```



```

    }];

}

- (IBAction)allplaces:(UIButton *)sender {
    iconname=nil;

    [_mapview removeAnnotations:[_mapview annotations]];

    PFQuery *query1 = [PFQuery queryWithClassName:@"Dangersplaces"];

    [query1 findObjectsInBackgroundWithBlock:^(NSArray *objectss, NSError *error) {
        if (!error) {
            CLLocationCoordinate2D annoo;
            iconnumber=objectss.count;
            for (int i=0; i<objectss.count; i++)

                { MyAnnotation *aaa=[[MyAnnotation alloc]init];

floatValue ];
                annoo.latitude=[[ [objectss objectAtIndex:i] objectForKey:@"latitude"]
floatValue];
                annoo.longitude=[[ [objectss objectAtIndex:i] objectForKey:@"longtitude"]

                aaa.coordinate=annoo;
                aaa.title=[ [objectss objectAtIndex:i] objectForKey:@"title"];
                aaa.subtitle=[ [objectss objectAtIndex:i] objectForKey:@"subtitle"];
                aaa.iconname=[ [objectss objectAtIndex:i] objectForKey:@"iconname"] ;

                [_mapview addAnnotation:aaa];

                }

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }

    }];

}

- (IBAction)maptype:(UISegmentedControl *)sender {
    if (maptype.selectedSegmentIndex==0) {
        _mapview.mapType=MKMapTypeStandard;

    }else if (maptype.selectedSegmentIndex==1) {
        _mapview.mapType=MKMapTypeHybrid;
    }else if (maptype.selectedSegmentIndex==2) {
        _mapview.mapType=MKMapTypeSatellite;
    }

}

-(void)viewDidAppear:(BOOL)animated
{

```

```

_mapview.showsUserLocation = YES;
_mapview.zoomEnabled = YES;
_mapview.scrollEnabled = YES;
_mapview.mapType = MKMapTypeStandard;
_mapview.delegate=self;
[self.table setDelegate:self];

[self.table setDataSource:self];
CLLocationCoordinate2D p;
p.latitude=45.550815;
p.longitude=-61.023674;
MKCoordinateRegion mapRegion;
mapRegion.center = p;
mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
[_mapview setRegion:mapRegion animated: YES];

myList= [NSMutableArray arrayWithObjects: nil, nil, nil, nil];
detail= [NSMutableArray arrayWithObjects: nil, nil, nil, nil];
PFQuery *query = [PFQuery queryWithClassName:@"Dangersplaces"];

[query whereKey:@"iconname" notEqualTo:@"erro"];

[query findObjectsInBackgroundWithBlock:^(NSArray *results, NSError *error) {
    for (int i=0; i<results.count; i++) {
        if ([results objectAtIndex:i] valueForKey:@"iconname"]==nil) {
        }else{

            NSArray *aa=[[results objectAtIndex:i] valueForKey:@"date" ] ;

            [detail insertObject:aa atIndex:i];

        }

    }

    NSSortDescriptor * sortDesc = [[NSSortDescriptor alloc] initWithKey:@"self"
ascending:NO];
    [detail sortUsingDescriptors:[NSArray arrayWithObject:sortDesc]];
    [self.table reloadData];

}];
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSectionSection:(NSInteger)section{
    return [detail count];
}

```

```

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *simpleTableIdentifier = @"addplaces";

    UITableViewCell *cell = [tableView
    dequeueReusableCellWithIdentifier:simpleTableIdentifier];

    if (cell == nil) {
        cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault
        reuseIdentifier:simpleTableIdentifier];
    }

    cell.textLabel.text = @"CreatedAt:";
    cell.detailTextLabel.text=[detail objectAtIndex:indexPath.row];

    return cell;
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath
*)indexPath {

    iconname=nil;
    name=@"start";

    [_mapview removeAnnotations:[_mapview annotations]];
    UITableViewCell *cell = [tableView cellForRowAtIndexPath:indexPath];

    PFQuery *query1 = [PFQuery queryWithClassName:@"Dangersplaces"];

    [query1 whereKey:@"date" equalTo:cell.detailTextLabel.text];

    [query1 getFirstObjectInBackgroundWithBlock:^(PFObject *object, NSError *error) {
        if (!error) {
            CLLocationCoordinate2D annoo;

            MyAnnotation *aaa=[[MyAnnotation alloc]init];

            annoo.latitude=[[ object objectForKey:@"latitude"] floatValue ];
            annoo.longitude=[[ object objectForKey:@"longtitude"] floatValue];

            aaa.coordinate=annoo;
            aaa.title=[ object objectForKey:@"title"];
            aaa.subtitle=[ object objectForKey:@"subtitle"];
            self.tablenotification.text=[ object objectForKey:@"notification"];
            self.tabletitle.text=[ object objectForKey:@"title"];
            self.tablesubtitle.text=[ object objectForKey:@"subtitle"];

            aaa.iconname=[ object objectForKey:@"iconname"] ;

            [_mapview addAnnotation:aaa];

        } else {
            // Log details of the failure
            NSLog(@"Error: %@ %@", error, [error userInfo]);
        }
    }
}

```

```

    }];

    // [self.table reloadData];

    /*UIAlertView *messageAlert = [[UIAlertView alloc]
        initWithTitle:@"Row Selected" message:@"You've selected a row" delegate:nil
        cancelButtonTitle:@"OK" otherButtonTitles:nil];

    [messageAlert show];*/
}

- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
{
    return YES;
}

- (void)tableView:(UITableView *)tableView
commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
forRowAtIndexPath:(NSIndexPath *)indexPath
{
    UITableViewCell *cell = [tableView cellForRowAtIndexPath:indexPath];

    if(editingStyle == UITableViewCellEditingStyleDelete)
    {
        PFQuery *query = [PFQuery queryWithClassName:@"Dangersplaces"];
        [query whereKey:@"date" equalTo:cell.detailTextLabel.text];
        [query getFirstObjectInBackgroundWithBlock:^(PFObject *object, NSError *error) {
            if (!object) {
                NSLog(@"The getFirstObject request failed.");
            } else {
                [object deleteInBackground];
                [object saveInBackground];
            }
        }];

        [_mapview removeAnnotations:[_mapview annotations]];
        [detail removeObjectAtIndex:indexPath.row];

        [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath]
        withRowAnimation:UITableViewRowAnimationLeft];
    }
}

- (UITableViewCellEditingStyle)tableView:(UITableView *)tableView
editingStyleForRowAtIndexPath:(NSIndexPath *)indexPath
{

```

```

        return UITableViewCellEditingStyleDelete;
    }

    - (IBAction)touch:(id)sender {
        name=nil;

    }

    - (void)handleLongPress:(UIGestureRecognizer *)gestureRecognizer
    {
        iconname=nil;

        if (name==nil) {
            red.hidden=NO;
            yellow.hidden=NO;
            green.hidden=NO;
            if (gestureRecognizer.state != UIGestureRecognizerStateBegan)
                return;

            [_mapview removeAnnotations: _mapview.annotations] ;

            CGPoint touchPoint = [gestureRecognizer locationInView:self.mapview];

            CLLocationCoordinate2D touchMapCoordinate =
            [self.mapview convertPoint:touchPoint toCoordinateFromView:self.mapview];

            coord=touchMapCoordinate;

            NSString *la= [NSString stringWithFormat:@"%f",touchMapCoordinate.latitude];
            NSString *lo= [NSString stringWithFormat:@"%f",touchMapCoordinate.longitude];
            NSString *coordinate= [la stringByAppendingString:lo];

            MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

            point.coordinate = touchMapCoordinate;
            point.title = @"Coordinate";
            point.subtitle = coordinate;

            [self.mapview addAnnotation:point];
        }else

        iconname=nil;

    }

    - (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
    <MKAnnotation>)Annotation
    {

        MyAnnotation *annoo=(MyAnnotation *)Annotation;

        MKPinAnnotationView *pinView = nil;

        if(Annotation != _mapview.userLocation)
        {

            if (name==nil) {

```

```

        static NSString *defaultPinID = @"com.invasivecode.pin";

        pinView = (MKPinAnnotationView *)[_mapview
dequeueReusableAnnotationViewWithIdentifier:defaultPinID];
        if ( pinView == nil )
            pinView = [[MKPinAnnotationView alloc] initWithAnnotation:Annotation
reuseIdentifier:@"currentloc"];
        pinView.pinColor = MKPinAnnotationColorPurple;
        pinView.canShowCallout = YES;
        pinView.animatesDrop = YES;

    }else{

        for (int i=0; i<iconnumber; i++) {

            if (iconname==nil)
            {

                MKAnnotationView *custview=[[MKAnnotationView alloc] init];

                NSString * name=annoo.iconname;
                NSString *png=@"red.png";
                NSString *all=[name stringByAppendingString:png];
                UIImage * img =[UIImage imageNamed:all];
                CGRect resizeRect;
                resizeRect.size.height = 40;
                resizeRect.size.width = 40;
                resizeRect.origin = (CGPoint){0.0f, 0.0f};
                UIGraphicsBeginImageContext(resizeRect.size);
                [img drawInRect:resizeRect];
                UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
                UIGraphicsEndImageContext();
                custview.image = resizedImage;
                custview.canShowCallout=YES;
                custview.enabled = YES;
                custview.multipleTouchEnabled = NO;
                return custview;
            }else if ([iconname isEqual:@"red"])
            {

                MKAnnotationView *custview=[[MKAnnotationView alloc] init];

                UIImage * img =[UIImage imageNamed:@"red.png"];
                CGRect resizeRect;
                resizeRect.size.height = 40;
                resizeRect.size.width = 40;
                resizeRect.origin = (CGPoint){0.0f, 0.0f};
                UIGraphicsBeginImageContext(resizeRect.size);
                [img drawInRect:resizeRect];
                UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
                UIGraphicsEndImageContext();
                custview.image = resizedImage;
                custview.canShowCallout=YES;
                custview.enabled = YES;
                custview.multipleTouchEnabled = NO;
                return custview;
            }else if ([iconname isEqual:@"yellow"])
            {

                MKAnnotationView *custview=[[MKAnnotationView alloc] init];

```

```

        UIImage * img =[UIImage imageNamed:@"yellow.png"];
        CGRect resizeRect;
        resizeRect.size.height = 40;
        resizeRect.size.width = 40;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }else if ([iconname isEqual:@"green"])
    {

        MKAnnotationView *custview=[[MKAnnotationView alloc] init];

        UIImage * img =[UIImage imageNamed:@"green.png"];
        CGRect resizeRect;
        resizeRect.size.height = 40;
        resizeRect.size.width = 40;
        resizeRect.origin = (CGPoint){0.0f, 0.0f};
        UIGraphicsBeginImageContext(resizeRect.size);
        [img drawInRect:resizeRect];
        UIImage *resizedImage = UIGraphicsGetImageFromCurrentImageContext();
        UIGraphicsEndImageContext();
        custview.image = resizedImage;
        custview.canShowCallout=YES;
        custview.enabled = YES;
        custview.multipleTouchEnabled = NO;
        return custview;
    }

    }

    }
    else
    {
        [_mapview.userLocation setTitle:@"I am here"];
        return pinView;
    }

}

- (void)mapView:(MKMapView *)mapView didUpdateUserLocation:(MKUserLocation *)userLocation
{
    if(haveAlreadyReceivedCoordinates) {
        return;
    }

    haveAlreadyReceivedCoordinates = YES;
    CLLocationCoordinate2D p;
    p.latitude=45.550815;
    p.longitude=-61.023674;
    MKCoordinateRegion mapRegion;
    mapRegion.center = p;
    mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
    [_mapview setRegion:mapRegion animated: YES];
}

```

```

}
- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end

```

AddareasView

```

#import <UIKit/UIKit.h>
#import <MapKit/MapKit.h>
#import <MobileCoreServices/MobileCoreServices.h>

@interface AddareasViewController : UIViewController<MKMapViewDelegate,
UITextFieldDelegate,UITableViewDelegate,
UITableViewDataSource,UINavigationControllerDelegate>
@property (strong, nonatomic) IBOutlet UITableView *table;
@property (strong, nonatomic) IBOutlet MKMapView *mapview;
@property (strong, nonatomic) IBOutlet UITextField *areanotification;
@property (strong, nonatomic) IBOutlet UITextField *areainfor;
@property (strong, nonatomic) IBOutlet UILabel *shownotification;
@property (strong, nonatomic) IBOutlet UILabel *showinfor;
@property (strong, nonatomic) IBOutlet UISegmentedControl *maptype;

@end

#import "AddareasViewController.h"
#import <MapKit/MapKit.h>
#import <Parse/Parse.h>
#import <UIKit/UIKit.h>
#import "MyAnnotation.h"

CLLocationCoordinate2D coords[0];
int a;

@interface AddareasViewController ()

@end

@implementation AddareasViewController
{
    NSMutableArray *myList;
    NSMutableArray *detail;

}
@synthesize table=_table;
@synthesize mapview=_mapview;
@synthesize areanotification=_areanotification;
@synthesize areainfor=_areainfor;
@synthesize shownotification=_shownotification;
@synthesize showinfor=_showinfor;
@synthesize maptype=_maptype;

- (IBAction)maptype:(UISegmentedControl *)sender {
    if (_maptype.selectedSegmentIndex==0) {
        _mapview.mapType=MKMapTypeStandard;

    }else if (_maptype.selectedSegmentIndex==1) {
        _mapview.mapType=MKMapTypeHybrid;
    }
}

```



```

        }else if (_maptype.selectedSegmentIndex==2) {
            _mapview.mapType=MKMapTypeSatellite;
        }
    }

- (void)viewDidLoad
{
    [super viewDidLoad];
    [self.table setDelegate:self];

    [self.table setDataSource:self];
    _mapview.delegate=self;

    UILongPressGestureRecognizer *lpgr = [[UILongPressGestureRecognizer alloc]
                                           initWithTarget:self
                                           action:@selector(handleLongPress:)];
    lpgr.minimumPressDuration = 0.5; //user needs to press for 2 seconds
    [self.mapview addGestureRecognizer:lpgr];

    myList= [NSMutableArray arrayWithObjects: nil, nil, nil, nil];

    detail= [NSMutableArray arrayWithObjects: nil, nil, nil, nil];
    PFQuery *query = [PFQuery queryWithClassName:@"Dangersareas"];

    [query whereKey:@"index" greaterThan:[NSNumber numberWithInt:0]];

    [query findObjectsInBackgroundWithBlock:^(NSArray *results, NSError *error) {
        for (int i=0; i<results.count; i++) {
            if ([[results objectAtIndex:i] valueForKey:@"info"]==nil) {
                // result is nil
            }else{
                NSArray *aa=[[results objectAtIndex:i] valueForKey:@"info" ] ;
                NSArray *bb=[[results objectAtIndex:i] valueForKey:@"date" ] ;

                [detail insertObject:aa atIndex:i];
                [myList insertObject:bb atIndex:i];
            }
        }

    }

    [self.table reloadData];
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPath {
    [_mapview removeOverlays: _mapview.overlays];

    UITableViewCell *cell = [tableView cellForRowAtIndexPath:indexPath];

    PFQuery *query = [PFQuery queryWithClassName:@"Dangersareas"];

    [query whereKey:@"date" equalTo:cell.detailTextLabel.text];
}

```

```

[query getFirstObjectInBackgroundWithBlock:^(PFObject *object, NSError *error) {
    if (!object) {
        NSLog(@"The getFirstObject request failed.");
    } else {
        // The find succeeded.

        NSArray *ff;
        ff= [ object objectForKey:@"latitude"];
        NSArray *hh;
        hh= [object objectForKey:@"longtitude"];
        a =[[ object objectForKey:@"index"] integerValue];
        self.shownotification.text=[ object objectForKey:@"notification"];
        self.showinfo.text=[ object objectForKey:@"info"];
        CLLocationCoordinate2D coordss[a];

        for (int i=0; i<a; i++) {
            coordss[i].latitude = [[ff objectAtIndex:i] floatValue];
        }
        for (int i=0; i<a; i++) {
            coordss[i].longitude=[[hh objectAtIndex:i] floatValue ];
        }

        MKPolygon *polygon;
        polygon = [MKPolygon polygonWithCoordinates:coordss count:a];

        [_mapview addOverlay:polygon];

        CLLocationCoordinate2D points[2];
        points[0]= CLLocationCoordinate2DMake(coordss[0].latitude,
coordss[0].longitude);
        points[1]= CLLocationCoordinate2DMake(coordss[a-1].latitude, coordss[a-
1].longitude);
        MKPolyline *polyline;
        polyline = [MKPolyline polylineWithCoordinates:points count:2];
        [_mapview addOverlay:polyline];
    }
}];

}

- (IBAction)cancel:(UIBarButtonItem *)sender {
    [self dismissViewControllerAnimated:YES completion:nil];
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSectionSection:(NSInteger)section{
    return [detail count];
}

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *simpleTableIdentifier = @"addareas";

    UITableViewCell *cell = [tableView
    dequeueReusableCellWithIdentifier:simpleTableIdentifier];

```

```

        if (cell == nil) {
            cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault
reuseIdentifier:simpleTableIdentifier];
        }

        cell.textLabel.text=[detail objectAtIndex:indexPath.row];
        cell.detailTextLabel.text=[myList objectAtIndex:indexPath.row];
        return cell;
    }

- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
{
    return YES;
}

- (void)tableView:(UITableView *)tableView
commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
forRowAtIndexPath:(NSIndexPath *)indexPath
{
    UITableViewCell *cell = [tableView cellForRowAtIndexPath:indexPath];

    if(editingStyle == UITableViewCellEditingStyleDelete)
    {
        PFQuery *query = [PFQuery queryWithClassName:@"Dangersareas"];
        [query whereKey:@"date" equalTo:cell.detailTextLabel.text];
        [query getFirstObjectInBackgroundWithBlock:^(PFObject *object, NSError *error) {
            if (!object) {
                NSLog(@"The getFirstObject request failed.");
            } else {
                [object deleteInBackground];
                [object saveInBackground];
            }
        }];

        [_mapview removeOverlays:[_mapview overlays]];
        [detail removeObjectAtIndex:indexPath.row];

        [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath]
withRowAnimation:UITableViewRowAnimationLeft];
    }
}

- (UITableViewCellEditingStyle)tableView:(UITableView *)tableView
editingStyleForRowAtIndexPath:(NSIndexPath *)indexPath
{
    return UITableViewCellEditingStyleDelete;
}

- (void)handleLongPress:(UIGestureRecognizer *)gestureRecognizer

```

```

{
    if (gestureRecognizer.state != UIGestureRecognizerStateBegan)
        return;

    CGPoint touchPoint = [gestureRecognizer locationInView:self.mapview];

    CLLocationCoordinate2D touchMapCoordinate =
    [self.mapview convertPoint:touchPoint toCoordinateFromView:self.mapview];

    NSString *la= [NSString stringWithFormat:@"%f",touchMapCoordinate.latitude];
    NSString *lo= [NSString stringWithFormat:@"%f",touchMapCoordinate.longitude];
    NSString *coordinate= [la stringByAppendingString:lo];

    MKPointAnnotation *point = [[MKPointAnnotation alloc] init];

    point.coordinate = touchMapCoordinate;
    point.title = @"Coordinate";
    point.subtitle = coordinate;

    [self.mapview addAnnotation:point];
    CLLocationCoordinate2D points[2];
    coords[a] = touchMapCoordinate;

    MKPolyline *polyline;
    if (a==0) {
        a++;
    }else
        if (a==1) {
            points[0]=coords[0];
            points[1]=coords[1];

            polyline = [MKPolyline polylineWithCoordinates:points count:2];
            [_mapview addOverlay:polyline];
            a++;
        }
        else{
            points[0]= CLLocationCoordinate2DMake(coords[a-1].latitude, coords[a-
1].longitude);
            points[1]= CLLocationCoordinate2DMake(coords[a].latitude,
coords[a].longitude);

            polyline = [MKPolyline polylineWithCoordinates:points count:2];
            [_mapview addOverlay:polyline];
            a++;
        }
}

}

- (void)mapView:(MKMapView *)mapView didSelectAnnotationView:(MKAnnotationView *)view{
    CLLocationCoordinate2D points1[2];
    CLLocationCoordinate2D points2[2];
    MKPolyline *polygon1;
    MKPolyline *polygon2;

    MKPointAnnotation *annot;
    annot=[_mapview.selectedAnnotations objectAtIndex:0];
    if (a==3) {

        points1[0]= CLLocationCoordinate2DMake(coords[0].latitude, coords[0].longitude);

```

```

        points1[1]= CLLocationCoordinate2DMake(coords[a-1].latitude, coords[a-
1].longitude);
        polygon1 = [MKPolyline polylineWithCoordinates:points1 count:2];
        [_mapview addOverlay:polygon1];

    }else if (a>3){

        [_mapview removeOverlay:[_mapview.overlays objectAtIndex:a-2]];

        points2[0]= CLLocationCoordinate2DMake(coords[0].latitude, coords[0].longitude);
        points2[1]= CLLocationCoordinate2DMake(coords[a-1].latitude, coords[a-
1].longitude);
        polygon2 = [MKPolyline polylineWithCoordinates:points2 count:2];
        [_mapview addOverlay:polygon2];

    }
}

```

```

- (IBAction)confirm:(UIBarButtonItem *)sender {
    if ([self.arenotification.text length]!=0 && [self.areasinfofor.text length]!=0 &&
a!=0) {
        PFObject *placeObject = [PFObject objectWithClassName:@"Dangersareas"];
        PFObject *placeObject2 = [PFObject objectWithClassName:@"notification"];
        NSNumber *index=[NSNumber numberWithInt:a];
        [placeObject setObject:index forKey:@"index"];
        [placeObject saveInBackground];

        NSString *notifica;
        NSString *info;

        notifica=self.arenotification.text;
        info=self.areasinfofor.text;

        [placeObject setObject:notifica forKey:@"notification"];
        [placeObject saveInBackground];
        [placeObject2 setObject:notifica forKey:@"area"];
        [placeObject2 setObject:@" " forKey:@"place"];
        [placeObject2 saveInBackground];

        [placeObject setObject:info forKey:@"info"];

        NSString *latitude[a];

        for (int i=0; i<a; i++) {
            latitude[i]=[NSString stringWithFormat:@"%f",coords[i].latitude];
        }

        NSArray *Latitude = [NSArray arrayWithObjects:latitude count:a];

        [placeObject setObject: Latitude forKey:@"latitude"];
        [placeObject saveInBackground];
        NSString *longitude[a];

        for (int i=0; i<a; i++) {
            longitude[i]=[NSString stringWithFormat:@"%f",coords[i].longitude];
        }
    }
}

```

```

    }

    NSArray *Longitude = [NSArray arrayWithObjects:longitude count:a];

    [placeObject setObject: Longitude forKey:@"longitude"];

    [placeObject saveInBackground];
    NSDateFormatter *dateForm = [[NSDateFormatter alloc] init];
    [dateForm setDateFormat:@"yyyy-MM-dd hh:mm:ss"];
    NSString *dateString = [dateForm stringFromDate:[NSDate date]];

    [placeObject setObject:dateString forKey:@"date"];
    [placeObject saveInBackground];

    [self dismissViewControllerAnimated:YES completion:nil];
} else {

    UIAlertView *message = [[UIAlertView alloc] initWithTitle:@"Check Steps"
                                                         message:@"Step 1: Longpress on
the map to draw area. Step 2: Input required textfiled. Step 3: Press done to submit."
                                                         delegate:nil
                                                         cancelButtonTitle:@"OK"
                                                         otherButtonTitles:nil];

    [message show];

}

}

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (void)viewDidAppear:(BOOL)animated{
    a=0;
    _mapview.showsUserLocation = YES;
    _mapview.zoomEnabled = YES;
    _mapview.scrollEnabled = YES;
    _mapview.mapType = MKMapTypeStandard;
    _mapview.delegate=self;
    [self.table setDelegate:self];

    [self.table setDataSource:self];
    CLLocationCoordinate2D p;
    p.latitude=45.550815;

```

```

        p.longitude=-61.023674;
        MKCoordinateRegion mapRegion;
        mapRegion.center = p;
        mapRegion.span = MKCoordinateSpanMake(0.3, 0.3);
        [_mapview setRegion:mapRegion animated: YES];
    }

- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id )annotation
{
    MKPinAnnotationView *pinView = nil;
    if(annotation != _mapview.userLocation)
    {
        static NSString *defaultPinID = @"com.invasivecode.pin";

        pinView = (MKPinAnnotationView *)[_mapview
        dequeueReusableAnnotationViewWithIdentifier:defaultPinID];
        if ( pinView == nil )
            pinView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
            reuseIdentifier:@"currentloc"];
        pinView.pinColor = MKPinAnnotationColorPurple;
        pinView.canShowCallout = YES;
        pinView.animatesDrop = YES;

    }
    else
        [_mapview.userLocation setTitle:@"I am here"];
    return pinView;
}

- (MKOverlayView *)mapView:(MKMapView *)mapView viewForOverlay:(id )overlay
{
    MKPolylineView *polyLineView = [[MKPolylineView alloc] initWithOverlay: overlay];
    polyLineView.strokeColor = [UIColor colorWithRed:71.0f/255.0f green:54.0f/255.0f
    blue:255.0f/255.0f alpha:1.0f];
    polyLineView.lineWidth = 2.0;

    return polyLineView;
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end

```